University of Hertfordshire

BACHELOR OF SCIENCE DEGREE/DEGREE WITH HONOURS IN MULTIMEDIA TECHNOLOGY

Final Year Project Report
Department of Electronic, Communication and Electrical Engineering

3 - Dimensional Display of the Trajectory of a Golf Ball

Report by
Cindy Yung

Supervisor
Mr. David Lee

Date
April 2006

University Of Hertfordshire

DECLARATION STATEMENT

I certify that the work submitted is my own and that any material derived or quoted from the published or unpublished work of other persons has been duly acknowledged (ref. UPR AS/C/6.1, Appendix I, Section 2 – Section on cheating and plagiarism)

Student Full Name: Cindy Yung

Student Registration Number: 02041627

Signed: ……

Date: 07 June 2006

University Of Hertfordshire

ABSTRACT

This report is part of a set of projects for state-of-the art golf driving range.

3-Dimensional graphic is becoming more advance in our culture which can display sharp 3D objects, animation movies, interactive functions and many more.

With 3-Dimensional application, most 3D applications had pre-defined view point or movement.

This report will demonstrate how high level such languages as VRML and Java can interact with each other, allowing freedom restriction to change the outcome result of a Golf Ball trajectory in a 3-Dimensional environment.

University Of Hertfordshire

ACKNOWLEDGEMENTS

I would like to say a special thank you to Mr. David Lee for his supervision, support and encouragement. I am grateful to him for his guidance, constant support and encouragement which made it possible for me to achieve all my goals and provided a quality learning experience.

A special thank you should be given to my client Mr Norman Lindsay for providing the ideas, related information and support throughout the project.

I would also like to thank my family and friends for supporting me, tolerating my long hours of unavailability during this project.

University Of Hertfordshire

Table of Contents

DECLARATION STATEMENT i
ABSTRACT i
ACKNOWLEDGEMENTS ii
Table of Contents iii
List of Figures viii
List of Tables x
List of Formula xi
GLOSSARY xii
1. Introduction 1
1.1 Project Aims 1
1.2 Project Objectives 2
1.3 Restrictions 2
1.4 Structure of Report 3
1.5 Author's background 3
2. Background Information 4
2.1 Introduction 4
2.2 Basic 3D Concepts 4
2.2.1 Right-Hand rule 5
2.3 Brief history of VRML 7
2.3.1 VRML 1.0 7
2.3.2 VRML 2.0 7
2.3.4 X3D the next generation of VRML 2.0 9
2.3.5 X3D Baseline 9
2.4 History of Java 10
2.4.1 Basic of Java 10
2.4.2 Typical Java Environment 10
2.5 What is Java 3D 12
2.5.1 What Java 3D Aims? 12
2.5.2 Advantages 12
2.5.3 Disadvantages 13
2.6 Brief history of Golf 13
2.6.1 History of Golf Ball 14
2.6.2 Standard weight of Golf Ball 15
2.6.3 Physics of Golf Ball 16
2.7 Research of 3D golf simulation websites 17
2.7.1 TopGolf 17
2.7.2 Sports-coach 18
2.8 Summary 19
3. Requirement Analysis 20
3.1 Introduction 20
3.2 User Requirements 20
3.2.1 Hardware 20

University Of Hertfordshire

3.2.2 Software....20
3.3 Developer Requirements....21
3.3.1 Languages....21
3.3.2 Software....21
3.3.3 Hardware....21
3.4 Summary....21
4. Methodology/Implementation....22
4.1 Introduction....22
4.2 Stages in choosing the suitable Languages....22
4.2.1 The First language tested....22
4.2.2 The Second language tested....22
4.2.3 The Third language tested....22
4.3 Preliminary findings....23
4.3.1 Background information on initial Implementation....23
4.3.2 VRML Data type....23
4.3.3 Text output into a text file "Buffered Reader"....24
4.3.4 Java IO class....24
4.4 Implementation of initial design....25
4.4.1 Sample section of initial application....25
4.4.2 Summary of initial application....25
4.5 Decision of what language to use....26
4.6 Flight calculation of a golf ball....26
4.6.1 Research on trajectory calculation....27
4.6.2 Euler's Forward method....27
4.6.3 Concern on trajectory formulas....27
4.6.4 Golf trajectory formulas....28
4.7 Implementation of the design version 2....28
4.7.1 Sample section of application version 2....29
4.7.2 Summary of the application version 2....29
4.8 Resolve error on application version 2....30
4.8.1 Additional formulas....30
4.9 Implementation of the design version 3....31
4.9.1 Sample section of application version 3....31
4.9.2 Summary of the application version 3....32
4.10 3D golf course 'VRML'....33
4.11 Software....33
4.11.1 VrmlPad....33
4.11.2 AC3D....33
4.11.3 NotePad....34
4.12 Implementation of VRML golf course....34
4.12.1 Sample section of the 3D golf course....34
4.12.2 Summary of VRML golf course....36
4.13 Screen shots of final VRML golf course....36
4.14 Final implementation....37
4.14.1 Separation of files methods....38
4.14.2 Sample section of Final application....39
4.14.3 Flow Chart Final application....39
4.14.4 Summary of the Final implementation....40
4.15 Summary....40
5. Testing....41

University Of Hertfordshire

5.1 Introduction 41
5.2 Test result: Application and Golf course 01 41
5.3 Test result: Application and Golf course 02 42
5.4 Test result: Application and Golf course 03 43
5.5 Summary 44
6. Conclusion 45

REFERENCES 47

BIBLIOGRAPHY 48

Useful Site For X3d 48
X3D and XML 49
X3D Example 49
X3D 49
Information on VRML 49
http://tecfa.unige.ch/guides/vrml/vrmlman/node5.html Proto 49
VRML Script 49
Xvrml Included Right Handed Rule 49
Java 3d And Vrml 97 49
VRML1.0 50
VRML Collision Node 50
VRML Eg:Collision Node 50
Object-To-Object Collision Detection Interface 50
VRML History 50
Good VRML History 50
Vrml1 Vs Vrml2 50
VRML Cortona Client 50
VRML Web3d 50
Vrml97 Web3d 50
VRML 1.0 50
VRML Browsers 50
VRML 97 Spec 51
VRML 2.0 Vs X3D 51
VRML SYATAX 51
MICROSOFT EXCEL 51
If Statement In Excel 51
Atan Excel 51
Sqrt Excel 51
Maths_Euler's Method Golf Ball 51
Golf Information 51
Dimples Vs Smooth Golf Ball 51
History of golf ball 52
Brief history of golf 52
Find The Right Ball For Your Game 52
Golf Ball 52
Golf 52
Golf Course 52
3D Golf Simulation 53
JAVA RESEARCH 53
Java 3d Site 53
Basic Java 53

University Of Hertfordshire

Buffer Replace....53
Read And Write A File....54
String Buffer....54
How to Use File Streams....54
Read a file and print, using BufferedReader and System.out....54
Reading Numbers from a Text File, Line by Line....54
3d World....54
Right Handed Rule....54
APPENDICES....55
Appendix A....55
Example of 3D languages....55
Java 3D....55
VRML 2.0....56
X3D....56
X3D Source code....57
Appendix B....58
Implementation of application versions 1-3....58
Version 1 – Initial application....58
Version 1 - GolfInput.java Source code....58
Version 1 - ReadWriteFile.java Source code....59
Version 1 - ReadWriteFile.java Source code....60
Version 2 - application....62
Version 2 - GolfInput.java Source code....63
Version 2 –ReadWriteFile.java Source code....65
Version 3 - application....70
Version 3 – GolfInpt.java Source code....71
Version 3 – ReadWriteFile.java Source code....73
Appendix C....79
Final application screen shot....79
Source Code....79
GolfInput.java - Interface....80
ReadWriteFile.java – The main program....83
Implemented structure of GolfInput.class....87
Source code - GolfInput.class....88
Source code - GolfInput$1.class....89
Source code - GolfInput$2.class....89
Implemented structure structure of ReadWriteFile.class....90
Source code - ReadWriteFile.class....90
Appendix D....91
Screen Shots of all VRML Golf courses....91
Golf Course 01....91
Golf Course 02....92
Golf Course 03....93
Appendix E....94
Golf course Multiple point views....94
Appendix F....95
Test Results (screen shots)....95

University Of Hertfordshire

Appendix G 96
User Manual 96
How to install and run '3D Golf Ball Trajectory' application 96
How to navigate around VRML browser 104
Moving around: Walk, Fly, and Study 104
Restore, Fit, and Align 109
Using Keyboard for Navigating 110
Appendix H 113
Questionnaire 113
Questionnaire results 114
Appendix I 116
Story board 116
Interface designs 116
Golf course designs 117
Appendix J 118
Project Time Plan – Gantt Chart 118
Initial Project Time Plan Gantt Chart 119
Final Project Time Plan Gantt Chart 120
Appendix K 125
CD-ROM Contents 125
Application 125
Final Report 125
Poster 125
Software 125
ReadMe.htm 125

University Of Hertfordshire

List of Figures

Figure 1: XY coordinates 4
Figure 2: XYZ coordinates 5
Figure 3: Right Hand Rule 6
Figure 4: Right-Hand Rule for rotation 6
Figure 5: X3D Baseline 9
Figure 6: Typical Java environment 11
Figure 7: Stages of Golf Ball 15
Figure 8: Spin Trajectory Of A Golf Ball 16
Figure 9: Dimples Range Of A Golf Ball 17
Figure 10: On the Left the out field of TopGolf layout and 18
Figure 11: LS Golf 3 simulator 19
Figure 12: Concept of initial application 23
Figure 13: Java IO class 24
Figure 14: Sample section initial application - ReadWriteFile.java 25
Figure 15: Design structure 26
Figure 16: Excel golf trajectory 28
Figure 17: Sample section of application version 2 – ReadWriteFile.java 29
Figure 18: Sample section of application version 3 – ReadWriteFile.java 31
Figure 22: Stage 1 36
Figure 23: Stage 2 36
Figure 24: Stage 3 36
Figure 25: Stage 4 36
Figure 26: Stage 5 36
Figure 27: Stage 6 36
Figure 28: Stage 7 36
Figure 29: Stage 8 36
Figure 30: Stage 9 36
Figure 31: Stage 10 36
Figure 32: Stage 10 36
Figure 33: Stage 12 37
Figure 34: FileWriter hierarchy [14] 38
Figure 35: Sample section of application version 3 – ReadWriteFile.java 39
Figure 37: Java3D example from Mac OSX 55
Figure 38: VRML example - static 56
Figure 39: VRML example - Animated 56
Figure 40: X3D – Animated robot 56
Figure 41: X3D example source code 57
Figure 42: Initial application (Version 1) 58
Figure 43: Application Version 2 62
Figure 44:Final application 79
Figure 45: Structure of GolfInput.class 87
Figure 46: Structure of GolfInput.class 90
Figure 47: Golf course 01 – Tee low view 91
Figure 48: Golf course 01 – Tee high view 91
Figure 49: Golf course 01 – Bird eye view 91
Figure 50: Golf course 02 – Tee low view 92
Figure 51: Golf course 02 – Tee high view 92

University Of Hertfordshire

Figure 52: Golf course 02 – Bird eye view....92
Figure 53: Golf course 03 – Tee low view....93
Figure 54: Golf course 03 – Tee high view....93
Figure 55: Golf course 03 – Bird eye view....93
Figure 56: Multiple view points....94
Figure 57: Mr. Norman Lindsay sign pole....95
Figure 58: Insert values in all text fields on 'Golf course 01'. Test on submission of values....95
Figure 59: Test if 'GolfTek' billboard open....95
Figure 60: Test if 'Lindsay Golf' billboard open....96
Figure 61: Interface design 01....116
Figure 62: Interface design 02....116
Figure 63: Interface design 03....116
Figure 64: Golf course design 01....117
Figure 65: Golf course design 02....117
Figure 66: Golf course design 03....117
Figure 67: Golf course design 04....117
Figure 68: Initial Gantt Chart....119

University Of Hertfordshire

List of Tables

Table 1: VRML Date type 23
Table 2: Outcome results version 2 29
Table 3: Outcome results version 3 32
Table 4: Outcome results Final application 40
Table 5: Test result: Application and Golf course 01 41
Table 6: Test result: Application and Golf course 02 42
Table 7: Test result: Application and Golf course 03 43

University Of Hertfordshire

List of Formula

Formula 1: Method to solve for velocity at time $t_1 = t_0 + \Delta t$.....27
Formula 2: Method for the components of position.....27
Formula 3: Output data.....28
Formula 4: Additional formulas.....30

University Of Hertfordshire

GLOSSARY

Java **compiler** (javac) translates a Java program into bytecodes- the language understood by the Java interpreter. If a program compiler produces a file with the .class extension. This is the file containing the bytecodes that are interpreted during the execution phase. [1]

A **Java program** must first be placed in memory before it can execute. This is done by the class loader, which takes the .class file (or files) containing the bytecodes and transfers it to memory. The .class file can be loaded from a disk on your system ir over a network. [1]

An **application** is a program that is normally stored and executed on the user's local computer. Applications are loaded into memory, then executed by the java interpreter. [1]

An **applet** is a program that is normally stored on a remote computer that users connect to via a Web browser. Applets are loaded from a remote computer into the browser to execute. [1]

Applet can also be executed from the command line using the **appletviewer** command provided with the Java 2 Software Development Kit J2SDK). The appletviewer is commonly referred to as the minimum browser- it knows only how to interpret applets. [1]

University Of Hertfordshire

1. Introduction

From the start it was obvious the project title "3-Dimensional Display of the Trajectory of a Golf Ball" will involve 3-Dimensional language such as VRML for the outcome display, some type of mathematics/calculation language such as Java for calculating the trajectory and Golf knowledge to understand overall of golf. This would require a lot of research to fully understand and achieve the project.

The project title brief is as follows:
This project is part of a set of projects for a state-of-the art golf driving range. Given the velocity and spin of the ball, this project will involve the production of a 3D animation of the trajectory of the ball, using tools such as VRML and JAVA. (A mathematical module will be provided to calculate the trajectory.) It should overlay the animation on an image of an existing golf course.

1.1 Project Aims

The project aims to develop a 3D simulation of a golf ball trajectory. When the ball is hit, an application will display a predication where the ball is going to land, the angle of the ball and backspin or sidespin.

This simulation will be as realistic / accurate as possible and this application is the beginning plan of a major project for my client called Mr Norman Lindsay. He is an inventor who aim to build a golf course with gadget / sensor imbedded. This golf course will be one of the most up to date in the world, of never ending changes of technology society.

University Of Hertfordshire

1.2 Project Objectives

The objective is to create an application that can predict where the ball is going to land, the angle of the ball and backspin or sidespin. Instead of the automated system, the end-users can use the application to get the results needed for their shots after the shots have been taken. For example, users can input values on speed, distance, vertical angle, backspin and sidespin. Once users select to take the shots, a 3D simulation will display the flight and rotation of the golf ball.

1.3 Restrictions

While working on the project, there were a number of restrictions that delayed the process of the simulation. The setbacks include the following:

- Knowledge of programming languages:
  VRML: Having to learn the language this year, it will require a lot of research into its usage and functions. This will take up a lot of time from the project in research and reading.
  Java: Would be the best language for controlling the calculation of the golf ball. With my knowledge of Java at a beginner level, it would have negative impact on the time required to get the standard necessary for this project.
- To create a simulation involving interaction and 3D movement, a suitable graphics card is needed.
- VRML browser: a software which can be downloaded for free
- Java J2SE kit: software can be downloaded for free

University Of Hertfordshire

1.4 Structure of Report

This report has been written for two types of readers, first type of readers being experienced technologists who are not familiar in this field and second type of readers being technologists who are specialists in this field of the project.

Readers that are experienced in this subject may wish to skip the initial chapters to gain rapid understanding of the project. Readers with brief or no knowledge of this subject are advised to read all chapters to fully understand the project.

1.5 Author's background

Author has studied Multimedia Technology at University of Hertfordshire for few years, and has a placement year working with highly trained engineers in technical support for hard and software. From the experience gained, this will be a challenging project as the author has basic knowledge of Java to create a multimedia application and no knowledge of VRML until this year. To fully meet the requirement of the aims and objectives of this project, it could be considered a restriction to the project but this can be resolved by familiarization with the languages in full ability.

University Of Hertfordshire

2. Background Information

2.1 Introduction

In this chapter it will describe each language that could be related to this project, information on golf ball and a summary of which language is best suited for the golf simulation.

2.2 Basic 3D Concepts

Let start with 2D graphic as most people are familiar with 2D graphic know as XY coordinates. *(Figure 1)*

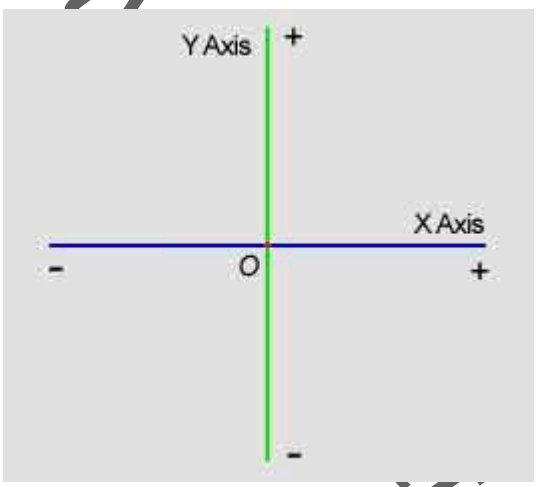


Figure 1: XY coordinates

(Source: xvrml.net)

The difference of 3D coordinates from 2D is that 3D contains three axis which is know as XYZ. To enable understanding of 3D world, there is a need to understand why there are three axis coordinates first.

With the illustration below *(Figure 2)* it should provide a better understanding on how 3D works.

University of Hertfordshire

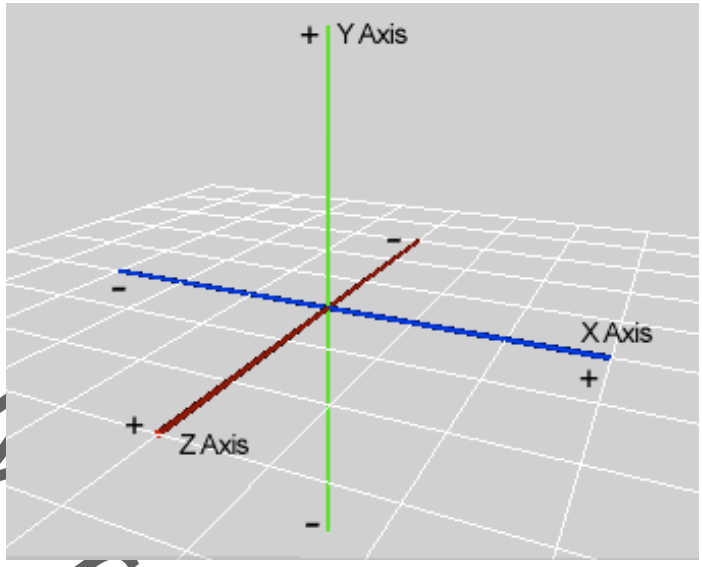


Figure 2: XYZ coordinates

(Source: xvrml.net)

The coordinate of point represents in a 3D space is X,Y,Z, the origin of coordinate is 0.0, 0.0, 0.0.

X axis : As seen from *Figure 2*, line from 'left' is negative to 'right' is positive, along the X axis the coordinate presents (**X**, 0.0, 0.0)

Y axis : As seen from *Figure 2*, line from 'down' is negative to 'up' is positive, along the Y axis the coordinate presents (0.0, **Y**, 0.0)

Z axis : As seen from *Figure 2*, line from 'far' is negative to 'close' is positive, along the Z axis the coordinate presents (0.0, 0.0, **Z**)

2.2.1 Right-Hand rule

Using the right hand rule it is to refer to the default axis in a 3D world which can also be used to identify positive rotation. The figures below will explain in more details.

University Of Hertfordshire

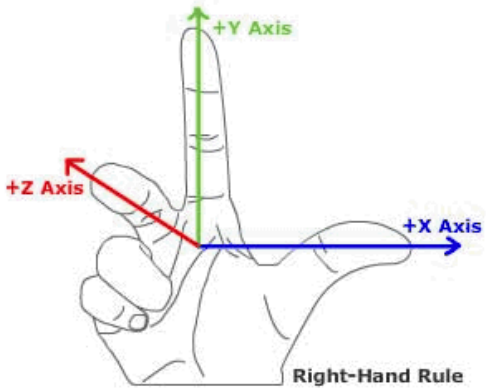


Figure 3: Right Hand Rule

(Source: xvrml.net)

Thumb: point to positive **X**

Index finger: point to positive **Y**

Middle finger: point to positive **Z**

In rotating an object, right-hand rule can help identify which direction to rotate. *(Figure 4)*

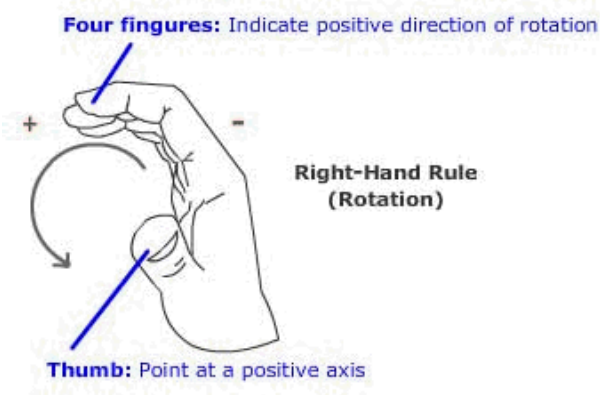


Figure 4: Right-Hand Rule for rotation

(Source: xvrml.net)

Thumb: point along a positive axis depends on which axis you are going to rotate.
Other four fingers: indicate the positive direction of rotation

(Note: In VRML angle of degrees need to convert to radius, the formula is as follows: number of degrees/360*3.14)

University Of Hertfordshire

2.3 Brief history of VRML

VRML stands for "Virtual Reality Modelling Language" which was born in 1994 as the first version VRML appeared to the world. VRML didn't take off to the world and the reason behind it was that:

- The slowness over the net, back in the days internet speed run at 28K or less, which took a long time to load
- VRML needs to run on specific browser in order to view 3D objects
- Can not mix / integrate with other programming languages

As from above the reason why VRML didn't take off as much as they hoped for was down to un-user friendly process.

2.3.1 VRML 1.0

The first version of VRML allows for the creation of virtual worlds with limited interactive behaviour. These worlds can contain objects which have hyper-links to other worlds, HTML documents or other valid MIME types. When the user selects an object with a hyper-link, the appropriate MIME viewer is launched. When the user selects a link to a VRML document from within a correctly configured WWW browser, a VRML viewer is launched. Thus VRML viewers are the perfect companion applications to standard WWW browsers for navigating and visualizing the Web. Future versions of VRML will allow for richer behaviours, including animations, motion physics and real-time multi-user interaction. [1]

2.3.2 VRML 2.0

VRML is a language for specific dynamic 3D scenes through which users can navigate with the help of a VRML browser. VRML scenes can be distributed over the

University Of Hertfordshire

World Wide Web with special VRML browsers that plug-ins to the Web browsers. VRML is well integrated with the WWW e.g. VRML scenes can be connected to any URLS.

VRML 2.0 is also known as VRML97. In comparison, VRML97 contains numerous changes to the VRML 1.0.

Here are the changes:

- Interactive objects
- Audio
- Behaviour
- Scripting

In this cases, VRML97 is far more complex than VRML 1.0.

To view VRML as mentioned, it needs VRML browser plug-ins to the web browser. Here are a list of browsers that can be downloaded for Windows platforms:

Cosmo Player: One of the more popular VRML plug-ins, although it is no longer being updated and is not being supported technically. Compliant with VRML97.
Openwrl: VRML97 viewer, with model's parts controlling (Lights, separate objects), importing one file to another.
Cortona VRML Client : Works as a highly interactive VRML viewer plug-in for popular Internet browsers and office applications. Complete VRML97 support, including Java and JavaScript support, hardware acceleration, stereoscopic rendering, automation interface.
FreeWRL: An open-source VRML and X3D browser written by a group of developers, who wish to produce a high quality, up to date, browser that is in the public domain. (O/S Mac)

University Of Hertfordshire

2.3.4 X3D the next generation of VRML 2.0

X3D in an Open Standards XML-enabled 3D file format to enable real-time communication of 3D data across all applications and network applications. It has a rich set of features for use in engineering and scientific visualization, CAD and Architecture, Medical visualization, Training and simulation, multimedia, entertainment, educational, and more. [2]

X3D is a considerably more mature and refined standard than its VRML predecessor so authors can achieve the behaviours they expect. [2]

2.3.5 X3D Baseline

- **Interchange** is the basic profile for communicating between applications. It support geometry, texturing, basic lighting, and animation.
- **Interactive** enables basic interaction with a 3D environment by adding various sensor nodes for user navigation and interaction (e.g. PlanseSensor, TouchSensor, etc.), enhanced timing, and additional lighting (Spotlight, PointLight).
- **Immersive** enables full 3D graphics and interaction, including audio support, collision, fog, and scripting.
- **Full** includes all defined nodes including NURBS, H-Anim and GeoSpatial components. [2]

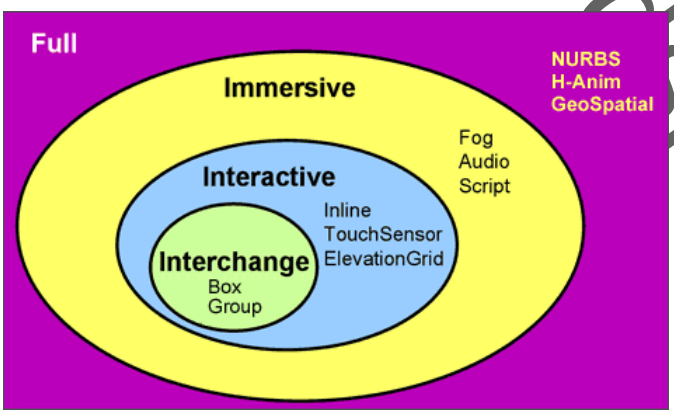


Figure 5: X3D Baseline

(Source: web3d.org)

University Of Hertfordshire

2.4 History of Java

Java was developed by Sun Microsystems. It operates on Java 2 Platform Standard Edition called the Java 2 Software Development Kit (J2SDK), version 1.4.1 which includes the minimum set of tools to write software in Java.

2.4.1 Basic of Java

Java is a powerful, cross-platform, object-oriented programming language suitable for writing anything from a distributed application that runs on a corporate network to a database-driver Web site to host your personal photo gallery. To make it easier to learn, the Java language was designed to resemble some of the most popular programming languages in use today, more notably C/C++ [2]

2.4.2 Typical Java Environment

In creating Java programs normally it goes through five phases to be executed using Java development environment *(Figure 6)*. These are edit, compile, load, verify and execute.

University Of Hertfordshire

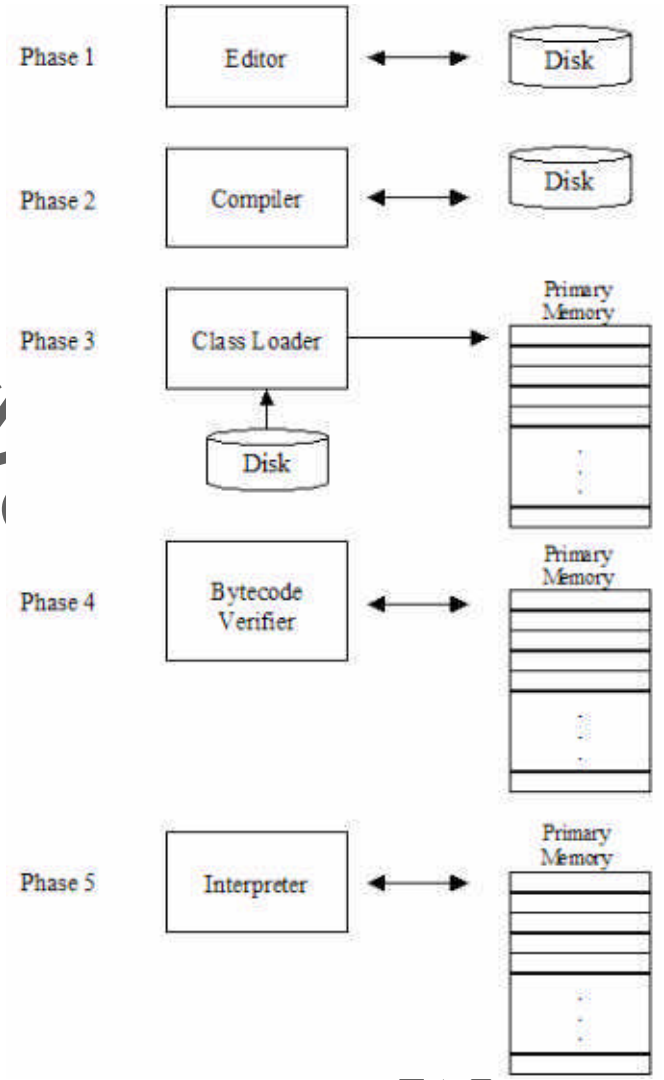


Figure 6: Typical Java environment

(Source: Java how to program by H.M Deitel and P.J. Deitel [4])

Phase 1: Program is created in an editor and stored on disk in a file ending with **.java**

Phase 2: Compiler creates btyecodes and stores them in disk in a file ending with **.class**

Phase 3: Class loader reads **.class** files containing bytecodes from disk and outs those bytecodes in memory.

Phase 4: Bytecode verifier confirms that all bytecodes are valid and do not violate Java's security restrictions.

Phase 5: Interpreter reads bytecodes and translates them into a language that the computer can understand possibly storing data values as the program executes.

2.5 What is Java 3D

Java 3D is an application programming interface (API) developed at Sun Microsystems for rendering interactive 3D graphics using the Java programming language. It usually runs on top of either Direct3D or OpenGL.

Comparing to other solutions Java 3D is not only a wrapper around these graphics API's but an interface that encapsulates the graphics programming using a real object-oriented concept. A scene could construct using a scene graph to represent the objects that have to be shown. Scene graph is structured as a tree containing several elements that are necessary to display the objects. Additionally Java 3D offers extensive specialized sound support. [5]

2.5.1 What Java 3D Aims?

When Java 3D was created, the aims are to create a high quality effect for 3D worlds with independent platform producing high performance in 3D graphics and with high-level 3D capacity for Java.

2.5.2 Advantages

What are the advantages in Java 3D, below are few advantages worth mentioning:

- Provide a high-level, object-oriented view of 3D graphics. Accomplished by scene graph based on 3D graphics model.
- A growing number of 3D loaders are available to import content into Java 3D runtime. For example: Java 3D VRML97 file loader and browser
- Java 3D support a number of interesting devices such as: wands, data gloves and headsets.

University Of Hertfordshire

- Java 3D is optimized for speed where possible. The runtime uses rendering capability bits, in fact, to optimize the scene graph for the faster possible renders, this approach makes Java 3D more applicable to interactive graphics environments (Games, simulations, low-latency situations) than to offline, high-quality graphics applications (like render farms) [6]

2.5.3 Disadvantages

As seen from above the advantages of Java 3D, so what are the disadvantages?

- Java 3D is a standard extension API, that mean programmers have to keep up to date with the additional changes to the core platform
- Java 3D had severe availability constraints. Look at OpenGL can be run on every type of operating system. In future cross-platform on Java 3D could be a problem
- Java 3D hides rendering-pipeline details from the developer. Because Java 3D is a high-level API, it intentionally hides details of the rendering pipeline from the developer, which makes it unsuitable for a significant number of problems where such details are important. [6]

2.6 Brief history of Golf

The very beginning of golf started in 1400's arguably in Scotland, later on the game itself was so popular (1457) that the King James II persuaded the Scottish Parliament to ban it! The reason behind this was King James II was worried that his men were spending too much time playing golf and not enough on practising their archery skills. The King felt that this would jeopardise Scotland in preparing to defend itself against an English invasion. The ban was reaffirmed in 1470 and 1491 even though many people largely ignored it.

University Of Hertfordshire

2.6.1 History of Golf Ball

When golf ball was in place, the game of golf needed to be changed such as: development of the golf club, golf course and the rules of the game were all affected by the Golf Ball evolution. Here are the stages of how the modern golf ball comes to live.

Feathery cube: Several pieces of stout leather were tightly stitched, leaving a small opening. The casing was turned inside out. Feathers - a "gentleman's top hat full" by measure - that had been boiled and softened, were tediously stuffed into the casing before the final stitches were made. The surprisingly hard feather ball was hammered into roundness and finally coated with several layers of "paint". Because of the difficulty and time involved, they were relatively expensive. This fragile missile was used for almost four centuries. [7]

The Gutta-Percha: The first "Gutta" ball is believed to have been made in 1848 by the Rev. Dr. Robert Adams Paterson from gutta-percha packing material. Gutta-percha is the evaporated milky juice or latex produced from a tree most commonly found in Malaysia. Gutta balls, were handmade by rolling the softened material on a board. The new durability of the Gutta, together with its much lower cost, resistance to water, and improved run, provided rejuvenation to the game of golf. [7]

The Hand Hammered Gutta: The hand hammered gutta was created by hammering the softened ball with a sharp edged hammer giving the ball an even pattern that greatly improved its play. Later, balls formed in iron moulds or ball presses that created patterns or markings on the ball were introduced. [7]

The Bramble: Surface textures and patterns impressed into the gutta-percha balls evolved from early imitations of feathery ball stitching to the highly detailed and symmetrical that greatly improved the ball's flight. The best known balls were the hand-marked private brands of the Scottish club makers, such as Morris, Robertson, Gourlay, and the Auchterlonies. Many brands with a variety of patent names used the

University Of Hertfordshire

bramble pattern (with a surface similar to the berry). This became the most popular pattern of the gutta era and was also used on some of the early rubber balls. [7]

The Rubber Ball: It was invented in 1898 by a Cleveland, Ohio, golfer, Coburn Haskell, in association with Bertram Work of the B. F. Goodrich Company. The ball featured rubber thread wound around a solid rubber core. The popular bramble, mesh, reverse mesh, and a great many other patterns gradually gave way to the aerodynamically superior dimple pattern first used in 1908. [7]

The Modern Ball: Durability and precision of today's ball reflect not only the tremendous technological advancement of their manufacture but also the development of space age plastics, silicone, and improved rubber. [7]

The image below show the stages of the golf ball in the following order (left to right): Feathery cube, The Gutta-Percha, The Hand Hammered Gutta, The Bramble, The Rubber Ball and The Modern Ball

Figure 7: Stages of Golf Ball

2.6.2 Standard weight of Golf Ball

In 1932 the standardization of golf ball weight and size was established by the United States Golf Assn (USGA).

The standardization of golf ball is as follows:

1. Weight: 1.620 ounces avoirdupois (45.93 gm).
2. Size: the diameter should not be less than 1.680 inches (42.67 mm).
3. Spherical Symmetry: The ball must not be designed, manufactured or intentionally modified to have properties which differ from those of a spherically symmetric ball.

University Of Hertfordshire

4. Initial Velocity: Should not exceed the limit specified by USGA, when measured on apparatus.
5. Overall Distance Standard: Golf ball should not exceed the distance specified under USGA conditions.

University Of Hertfo

2.6.3 Physics of Golf Ball

Imagine when playing golf and ever wonder how a golfer would know where to hit a ball to predict where it might land? With the explanation below it should give a better understanding when taking a shot in golf.

Let start of with backspin, when backspin imparted on a golf ball this will cause the ball gaining loft and stay in the air longer. As the ball spins in the air, the air above the ball is moving faster than the air below the ball. Because of the difference in air pressure, the ball itself has additional force pushing it up, which provides it to stay longer in the air. With the spin it also assists golfers to produce more accurate shots.

With a backspin imparted on a ball, the ball drops quicker toward the end of the shot. After hitting the ground, the backspin aid the ball to slow quicker. Here is a example of a ball-path with different amounts of backspin:

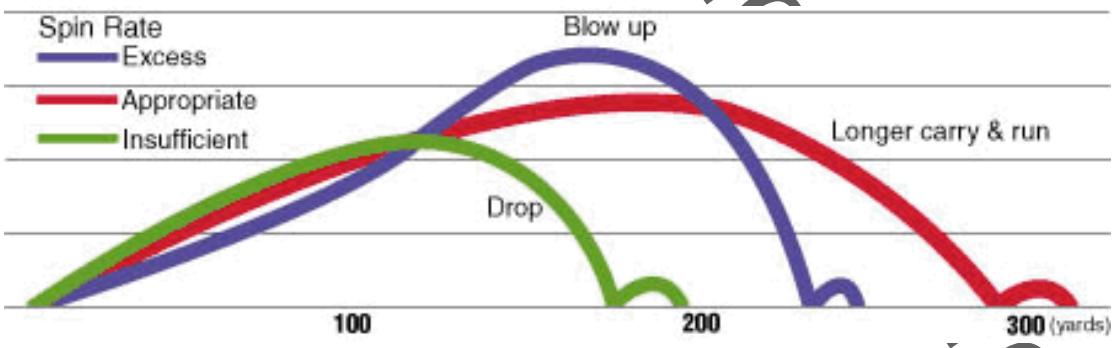


Figure 8: Spin Trajectory Of A Golf Ball

(Source ffden-2.phys.uaf.edu)

Dimpled golf balls fly much further than smooth balls. The reason why dimpled balls travel further than smooth balls is because the dimples on a golf ball create turbulence in the boundary layer (Boundary layer is a thin layer of air surrounding the golf ball as

it flies). This is helpful because the dimples would scoop the air back towards the rear of the ball. By moving more air to the rear, it prevent the air pressure to drop behind the ball. By doing this, the amount of air pressure pulling backwards on the ball is decreased. A diagram to show the affect of dimples with varying depths on a golf ball's distance:

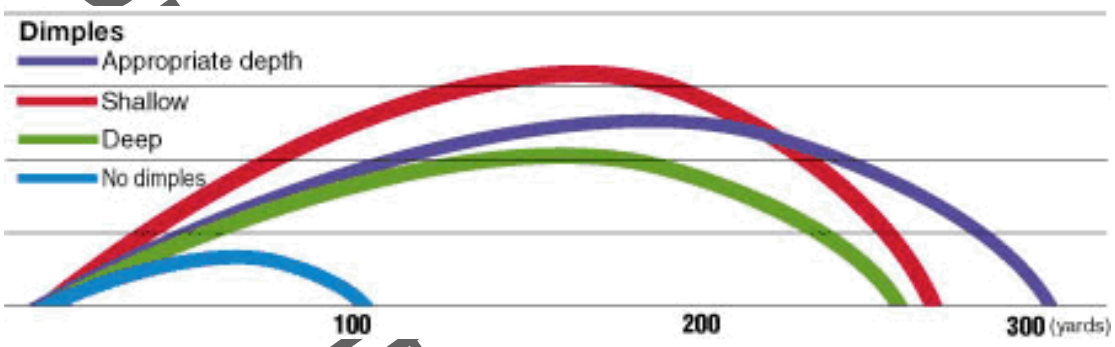


Figure 9: Dimples Range Of A Golf Ball
(Source ffden-2.phys.uaf.edu)

2.7 Research of 3D golf simulation websites

Various websites were examined with an aim to collect information, ideas and layout so as to analyse and produce top of the range 3D golf simulation. Of course at this moment there is no such web site existing over the internet, it is purely gathering information to produce a successful 3D golf simulation.

2.7.1 TopGolf

http://www.topgolf.co.uk/chigwell.htm

TopGolf is a point-scoring game with high-tech driving range. Aim to score as many points as possible by hitting chipped golf balls into any eleven targets on a ten acre outfield. After each shot, an exact distance ball travelled and the score is instantly relayed to a screen in your bay.

These are the stages on how TopGolf works:

University Of Hertfordshire

Step 1: Player collect golf ball from a ball dispenser. The I.D balls are automatically allocated to the players as the balls are dispensed.

Step2: Players register their ball in the bay to begin the game.

Step 3: Aim for selected target and strike ball from the mat

Step 4: Objective score as many points as possible. More points are awarded for hitting target further away and nearer to the centre target.

Step 5: Ball lands on target and passes through the I.D sensor

Step 6: The result instantly shows on display, showing distance the ball carried, where is landed and the number of point achieved.

Step 7: End of game, total scores of each player will be displayed and automatically uploaded to Game Centre leader board.

Figure 10: On the Left the out field of TopGolf layout and On the Right David Lee taking a shot in TopGolf

2.7.2 Sports-coach

http://www.sports-coach.com/index.aspx

Sport coach is a Golf simulator centre providing the latest golf simulation. Here is one of the simulator Sport coach support.

LS Golf 3 Simulator (Microsoft) has taken many years to develop to give golfers the realism they have demanded from playing indoor golf. Improved graphics, sound effects, course settings and weather settings, makes the LS Golf 3 a true test for any

University of Hertfordshire

golfer, whatever their handicap. Using their own clubs and balls they can now choose from a library of 38 Championship Courses, to test their skill.

Figure 11: LS Golf 3 simulator

Gathering from both websites, the layout of golf courses must be fun with some challenging difficulties. User friendly in the technical side of build, such as the application must be easy to access, clear display of golf ball trajectory and navigation around the application.

2.8 Summary

Let's recap on Norman Lindsay's idea. In his idea, sensors are imbedded underground over the whole golf course with wireless equipment to pick movement of each ball , which then instantly displays where the ball is landed (in prediction). So when golfer takes the shot, after 2-3 second a simulation will display on screen showing where the ball is landed and how distance travelled.

On that note the languages most suitable for this project are VRML 2.0 and Java 2 Software Development Kit (J2SDK). Why? Because both languages are free to download and minimum requirement is needed, spending less time in research means more time on the actual project.

University Of Hertfordshire

3. Requirement Analysis

3.1 Introduction

This chapter contains two main sections i.e. User requirements and Developer requirements. In those sections, each will provide relative requirements for different type of users. At the end of this chapter users should achieve the highest quality in view of the application.

3.2 User Requirements

For best view of the application, user are advised to follow the specification below to receive smooth running on the application.

3.2.1 Hardware

- Operating system: Microsoft Windows 98/ME/2000/XP or Windows NT 4.0.
- Processor: Pentium(r) 90 MHz or a faster processor.
- Random Access Memory (RAM): minimum of 128 MB.
- Display: SVGA/256 colour. The 800x600 high colour mode is recommended.
- Graphic card that support OpenGL or DirectX
- Input devices: Mouse and keyboard, joystick optional.

3.2.2 Software

- Internet Explorer 5.0 or a later version, Netscape Navigator 4.0 or a later version, Mozilla 1.5 or a later version, or Opera 7.0 or a later version.
- VRML browser can be downloaded free on http://www.parallelgraphics.com/products/cortona/download/iexplore/
- Java 2 Platform, Standard Edition (J2SE) can be downloaded free on http://java.sun.com/j2se/1.5.0/download.jsp

University Of Hertfordshire

3.3 Developer Requirements

To achieve the best outcome for this project, the following languages, software and hardware are needed to create this application.

3.3.1 Languages

- Java 2 Software Development Kit (J2SDK)
- VRML version 2.0

3.3.2 Software

- TextPad 4.7. or a later version
- VrmlPad 2.1 or a later version
- NotePad 5.1 or a later version
- Borland JBuilder 2005 or a later version
- AC3D 5.0 or a later version
- 3D studio Max 7 or a later version
- Adobe PhotoShop 7 or a later version
- VRML browser plug-in

3.3.3 Hardware

- O/S Window XP or a later version
- Intel Pentium 4.2.0GHz
- Minimum 128MB RAM
- Graphic card OpenGL or DirectX

3.4 Summary

This chapter is paying attention on user and developer requirements. With the requirements listed, this will provide a high-quality view of the 3D golf simulation.

Note: Hardware is noted to its minimum specification, with a more powerful machine the better it is.

University Of Hertfordshire

4. Methodology/Implementation

4.1 Introduction

In this section, the design and implementation process of this project are explained in details.

The design phase is the most important part of this project, in order to complete the project, most of the time was spent on research and planning. A carefully planned design will ensure the smooth implementation and future upgrade by examining all different scenarios that can happen.

4.2 Stages in choosing the suitable Languages

Testing different types of 3D languages that can be interacted in a 3D scenes should firstly be started.

4.2.1 The First language tested

Java 3D tried an example from "Java 3D Programming, By Daniel Selman" [8] It gives an animated scene, including a graphical textured background with directional lighting *(outcome example can be found in Appendix A)*.

4.2.2 The Second language tested

VRML 2.0 creates a .wrl file which contains a touch sensor to show that VRML can interact with end-user. The uncertainty on this language is whether it can take dynamic input. *(outcome example can be found in Appendix A)*.

4.2.3 The Third language tested

X3D as mentioned on Chapter 2.2.3 X3D is the next generation of VRML. It can do much more than VRML, the downside is author has no experience of this language. Thus, it may slow-down the process on the project. *(outcome example can be found in Appendix A)*

University Of Hertfordshire

4.3 Preliminary findings

To find out which languages are suitable for this project, an initial application will be created by using Java, JSP, web browsers and VRML. Below is figure 12 showing the concept of the initial application.

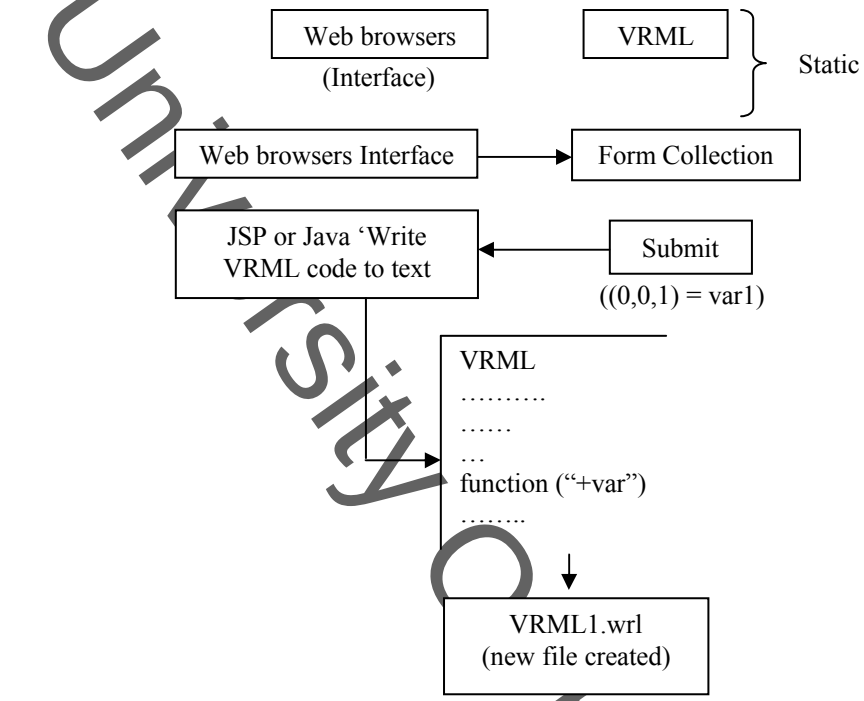


Figure 12: Concept of initial application

4.3.1 Background information on initial Implementation

Before building the initial application, there is a need to find out if VRML supports int, float, double etc, What kind of Java IO class is needed, and text output into a text file (in java) should also be looked at.

4.3.2 VRML Data type

The table below shows VRML does support int, float, double.

Table 1: VRML Date type

VRML Field Type	SFField Data Type	MFField Python Data Type
[SF/MF]Int32	int	Numeric-Python int array
[SF/MF]Float	float	Numeric-Python double array
[SF/MF]String	string (interpreted as UTF-8 unicode)	list of strings

University of Hertfordshire

[SF/MF]Time	float (time-module float)	Numeric-Python double array
SFBool	int (0/1)	
[SF/MF]Vec2f	2-item Numeric-Python double array	x*2 Numeric-Python double array
[SF/MF]Color, [SF/MF]Vec3f	3-item Numeric-Python double array	x*3 Numeric-Python double array
[SF/MF]Rotation	4-item Numeric-Python double array	x*4 Numeric-Python double array
SFImage	Numeric-Python int array	

4.3.3 Text output into a text file "Buffered Reader"

Each read request is made from a Reader. A Reader creates a corresponding read request to be added on the underlying character or byte stream. In this case BufferedReader cab warp around on any Reader whose read() operations could be costly such as, FileReaders and InputStreamReaders.

Example:

BufferedReader in

= new BufferedReader(new FileReader("name"));

4.3.4 Java IO class

File reader is used for input of character data from a disk file. The input file can be an ordinary ASCLL, one byte per character text file. A Reader stream automatically translates the characters from the disk file format into the internal characters format. The character in the input file might be from other alphabets supported by the UTF format, in which case there will be up to three bytes per character. In this case, characters from the file are translated into character format.

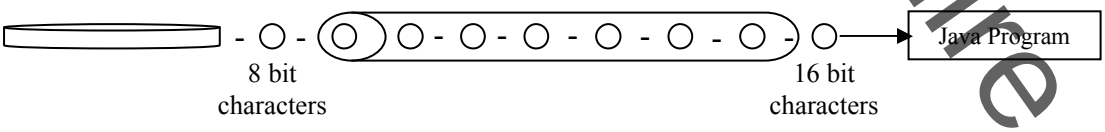


Figure 13: Java IO class

University Of Hertfordshire

4.4 Implementation of initial design

With the finding on research, now is time to put the theory in practice. The idea is to build the application using the VRML example mentioned in Chapter 4.1.2 to display the outcome, with two text fields in java for changing the VRML value
The end result should contain a NEW file called "Golf.wrl", a Java interface with input data and result of the submission from the interface, which display the new VRML file (Golf.wrl).

4.4.1 Sample section of initial application

```
import java.io.*;

public class ReadWriteFile {

 public ReadWriteFile(){
 }

 public ReadWriteFile(int val, int val2){
  setContent(val, val2);
 }

 static public void setContent(int Pos1, int Pos2)
 {
  System.out.println("New input ++++++++++++++++++ " + Pos1 );
  System.out.println("New input 2 +++++++++++++++ " + Pos2 );
  try {
     //FileWriter = new FileWriter
     BufferedWriter out = new BufferedWriter(new FileWriter("c:\\temp\\+Golf.wrl",
false));

     out.write(" #VRML V2.0 utf8\n\"n +……);
  }
out.close();
} catch (IOException e){
    }
  }
}
```

Key value to be changed in the VRML file

Specify the location of new file to be written to.

The content of the original VRML file

Figure 14: Sample section initial application - ReadWriteFile.java

4.4.2 Summary of initial application

The outcome of the application was successful. The outcome stages are as follows:

1. Java read the VRML file which is imbedded within itself
2. Using the Java interface, users are requested to input two values (XY axis) as they wishes.
3. The values would then be collected and print to be generated in a new file called 'Golf.wrl'

University of Hertfordshire

4. Open 'Golf.wrl' at the specific location, to display the outcome effect
5. Re-enter values on the interface and press refresh on the Internet browser to display new results.

As shown, this application is to prove the concept. With VRML imbedded inside java, java interface calls for the value change and displays the outcome result in VRML. *Screen shots of the initial application are placed in Appendix B.*

4.5 Decision of what language to use

After a successful implementation, decision is made to select the languages VRML and Java, as being proven both languages work well with each other which will also be using the tested structure *(Chapter 4.2.5)* to build the final product. The design structure is shown in Figure 15 below.

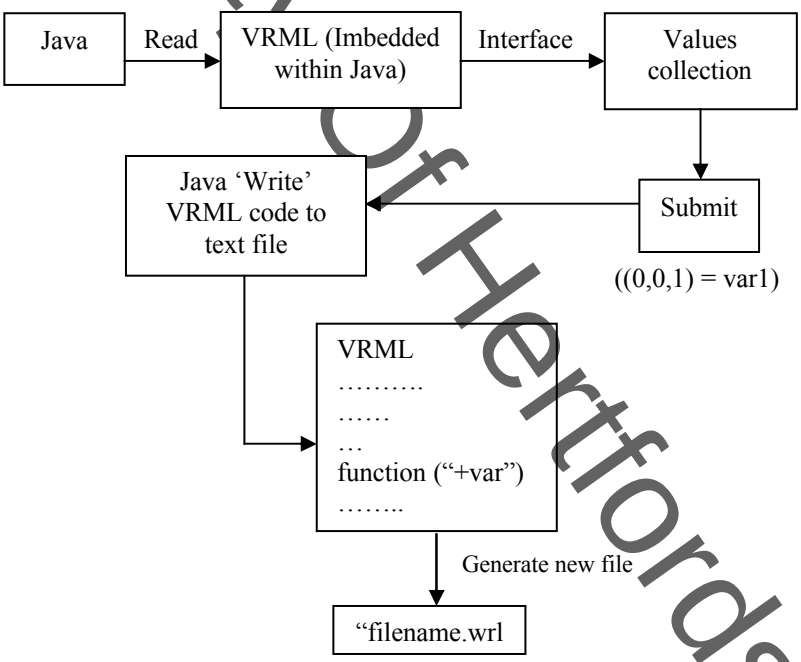


Figure 15: Design structure

Note: Design structure can be updated at anytime through out the project.

4.6 Flight calculation of a golf ball

After confirming that Java and VRML work well together, the next step is the mathematic formula of the golf ball flight movements.

University of Hertfordshire

This section is divided into two sub-sections, which include Research on trajectory calculation and Euler's Forward method.

4.6.1 Research on trajectory calculation

To achieve an accurate simulation, there is a need to understand the physics of golf ball flight pattern to calculate different movement, with the help from Euler's Forward Method.

4.6.2 Euler's Forward method

Euler's Forward Method is simple and flexible once the logic behind is understood. The advantage of Euler's Forward Method is that it can be used in Microsoft Excel which makes calculation of the trajectory easy. The common forms of Euler's Forward Method are described below:

Formula 1: Method to solve for velocity at time $t_1 = t_0 + \Delta t$

$v_x(t_1) = v_x(t_0 + \Delta t) \approx v_x(t_0) + (dv_x(t_0)/dt)\Delta t$

$v_y(t_1) = v_y(t_0 + \Delta t) \approx v_y(t_0) + (dv_y(t_0)/dt)\Delta t$

Formula 2: Method for the components of position

$x(t_1) = x(t_0 + \Delta t) \approx x(t_0) + v_x(t_0)\Delta t$

$y(t_1) = y(t_0 + \Delta t) \approx y(t_0) + v_y(t_0)\Delta t$

The method is going through *'Δt'* which uses the information from the beginning of the interval 'value *t0'*. The method itself is not perfect but like Eric W. Weisstein once said "The accuracy is actually not too bad and the stability turns out to be reasonable as long as the so-called Courant-Friedrichs-Lewy condition is fulfilled."[8]

4.6.3 Concern on trajectory formulas

With the research above, it proves that a lot of mathematical formulas are needed to be constructed in order to build a successful simulation. Due to lack of mathematical knowledge in the area of physics trajectory pattern, it leads to delay on the progress of the project.

As a result, Mr. N Lindsay has provided an Excel spreadsheet containing the answer to the calculations. From the spreadsheet, the numbers in red are the input data from

University Of Hertfordshire

the launch analyser *(figure 16)*, the output data formulas can be located on the first three columns, starting from row 10, which will then be fed into the simulation.

	A	B	C	D	E	F	G	H	I
1	**Golf Ball Trajectory**								
2	Columns A, B and C (row 10 onwards) give the ball position in metres at 0.1 second intervals.								
3	Input data in col. E			Speed (m/s)	55.00				
4				Vertical angle (deg)	16.00	(Always positive)			
5				Right/left (deg)	-3.00	(NB right is positive)			
6				Backspin (revs/sec)	60.00	(Always positive)			
7				Sidespin (revs/sec)	-20.00	(NB hooking spin is positive)			
8									
9	Distance	Height	Right/left	Va	Vb	Vc	V	alpha	
10	0.00	0.00	0.00	52.80	15.16	-2.77	55.00	-0.32	
11	5.28	1.52	-0.28	51.12	14.89	-2.34	53.29	-0.32	
12	10.39	3.00	-0.51	49.54	14.57	-1.95	51.67	-0.32	
13	15.35	4.46	-0.71	48.05	14.20	-1.59	50.13	-0.32	
14	20.15	5.88	-0.86	46.64	13.80	-1.25	48.66	-0.32	
15	24.81	7.26	-0.99	45.32	13.36	-0.95	47.26	-0.32	
16	29.35	8.60	-1.08	44.07	12.90	-0.66	45.92	-0.32	
17	33.75	9.89	-1.15	42.88	12.40	-0.40	44.64	-0.32	
18	38.04	11.13	-1.19	41.77	11.89	-0.16	43.43	-0.32	
19	42.22	12.32	-1.21	40.71	11.35	0.06	42.26	-0.32	
20	46.29	13.45	-1.20	39.71	10.80	0.27	41.15	-0.32	
21	50.26	14.53	-1.17	38.76	10.23	0.47	40.09	-0.32	
102									
103	NOTES								
104	Speed is in the range 20m/s for a short chip to 75m/s for a long drive								
105	Long shots (> 60m/s) have usually less than 15 degrees of vertical angle								
106	High lofted shots (> 20 degrees vertical angle) usually have speeds of less than 50m/s								
107	Backspin is roughly proportional to the vertical angle								
108		Assume: Backspin = 5*Speed*sin(E4/57) +50% to -50%							
109	Sidespin can be positive (hooking) or negative (slicing) and usually less than 50% of backspin								
110									
111									

Figure 16: Excel golf trajectory

4.6.4 Golf trajectory formulas

These are the formulas gathered from the spreadsheet which are shown in *formula 3*. The formulas will then be translated into Java which will be implemented into application version 2 “ReadWriteFile.java”.

Formula 3: Output data

$Va = x \text{ B } \cos\left(\frac{\text{[illegible]}}{57.296}\right) \text{ B } coz\left(\frac{\text{[illegible]}}{57.296}\right)$	*Pervious value = Prev*
$Vb = x \text{ B } \sin\left(\frac{\text{[illegible]}}{57.296}\right)$	*Distance = PreVa + 0.1 B Va*
$Vc = x \text{ B } \sin\left(\frac{\text{[illegible]}}{57.296}\right) \text{ B } \cos\left(\frac{\text{[illegible]}}{57.296}\right)$	*Height = PreVb + Vb B 0.1*
	Right or Left = PreVc + Vc B 0.1

4.7 Implementation of the design version 2

By using the same principle as ‘version1’, version 2 should enclose:

University Of Hertfordshire

- A Java interface with three text fields including: Speed (m/s), Vertical angle (deg) and Right/left (deg)
- Java application together with the trajectory calculations
- New .wrl file to be generated (.wrl is extension of VRML)
- A VRML file (golf.wrl) to display the outcome effect
- The outcome effect should be a smooth curve pattern of the golf ball trajectory

4.7.1 Sample section of application version 2

This sample contains all the mathematical formulas in Java

```
System.out.println("New input ++++++++++++++++ " + speed );
System.out.println("New input 2 ++++++++++++++ " + vAngle );
System.out.println("New input 3 ++++++++++++++ " + rightLeftVal
```

```
// State and collect previous Distance, Height and Right/left values
for(int i=0; i<100; i++)
        {
          distance = calDistance(speed, vAngle, rightLeftVal);
          height = calHeight(speed, vAngle);
          rightLeft = calRightLeft(speed, vAngle, rightLeftVal);

          preValA = preValA+0.1*distance;
          preValB = preValB+height*0.1;
          preValC = preValC+rightLeft*0.1;

          if(i<99)
            sb.append(" "+preValA+" "+preValB+" "+preValC+", \n ");
          else
            sb.append(" "+preValA+" "+preValB+" "+preValC+"] ");

          System.out.println("Current Pos "+i+" distance = "+preValA+"
          height = "+preValB+
                    " Right/Left = "+ preValC);
        }
```

Distance

```
public static double calDistance(double x, double y, double z)
{
  double dist = 0.00;

  dist = x*Math.cos(y/57.296)*Math.cos(z/57.296);
  return(dist);
}
```

Height

```
public static double calHeight(double x, double y)
{
  double height = 0.00;

  height = x*Math.sin(y/57.296);
  return(height);
}
```

Right/left

```
public static double calRightLeft(double x, double y, double z)
{
  double right = 0.00;

  right = x*Math.sin(z/57.296)*Math.cos(y/57.296);
  return(right);
}
```

Figure 17: Sample section of application version 2 – ReadWriteFile.java

4.7.2 Summary of the application version 2

The outcome was unsuccessful as the expected result was not matched.

Table 2: Outcome results version 2

Expected result	Actual result
A Java interface with three text fields including: Speed (m/s), Vertical (deg) and Right/left (deg)	Three text fields displayed: Speed (m/s), Vertical (deg) and Right/left (deg)
Java application together with the trajectory calculations	Calculations are in place
New .wrl file to be generated (.wrl is	New VRML file called ‘golf.wrl’ was

University of Hertfordshire

extension of VRML)	generated
A VRML file (golf.wrl) to display the outcome effect	Outcome effect displayed
The outcome effect should be a smooth curve pattern of the golf ball trajectory	Ball flied in a diagonal pattern of never ending height (no curve was displayed)

As seen from *table 2* the overall outcome was unsuccessful. To resolve the issue, the solution is back tracking into the steps, and then to work out what had gone wrong.

Screen shots of the application version 2 are placed in Appendix B.

4.8 Resolve error on application version 2

After investigation, a different formula on row 11 has been found in the worksheet *(figure 16)*. Previously the entire worksheet was not checked as the formulas are assumed to be the same through out the worksheet.

Also by changing the input data (the number in red) the entire worksheet will be dynamically updated with the results.

After the discovery, the next step is to interpret the finding into another design which will show how the formulas can create a smooth trajectory.

4.8.1 Additional formulas

These are the additional formulas that were missed on application version 2.

Formula 4: Additional formulas

$$SQRT\ VA \cdot VA + VB \cdot VB + VC \cdot VC$$

$$ATAN\ \frac{[illegible]}{Backspin}$$

$$VA\ \ VA + 0.0019 \cdot VB^{2} - 0.25 \cdot VA + 0.2 \cdot VC \cdot Alpha + VB^{2} \cdot IF\ \ Height < 0,0,1$$

$$VB\ \ VB + 0.98 + 0.0019 \cdot VB^{2} - 0.25 \cdot VB + 0.2 \cdot VA^{2} \cdot IF\ \ Height < 0,0,1$$

$$VC\ \ VC + 0.0019 \cdot VB^{2} - 0.25 \cdot VC + 0.2 \cdot VA \cdot Alpha^{2} \cdot IF\ \ Height < 0,0,1$$

SQRT: The SQRT function returns the square root of a number.

University Of Hertfordshire

ATAN: Returns the arctangent, or inverse tangent, of a number. The arctangent is the angle whose tangent is number. The returned angle is given in radians in the range $@_{2}^{\text{ﾠ}} to \frac{\text{ﾠ}}{2}$.

4.9 Implementation of the design version 3

Due to previous implementation of version 2 which didn't use all the formulas that were available on the spreadsheet, as a result the outcome was unsuccessful . Using version 2 as the base, the improvement of version 3 includes the following:

- Two extra text fields will be implemented into the interface. In total, there are five text fields : Speed (m/s), Vertical angle (deg), Right/left (deg), Blackspin (revs/sec) and Sidespin (revs/sec)
- The additional formulas will be implemented into the Java application
- A new .wrl file to be generated (.wrl is extension of VRML)
- The outcome effect should be a smooth curve of the golf ball trajectory

4.9.1 Sample section of application version 3

The sample section below shows the additional formulas added into Java

```
//initial calculation
vA = calInitialVa(speed, vAngle, rightLeftVal);
vB = calInitialVb(speed, vAngle);
vC = calInitialVc(speed, vAngle, rightLeftVal);
alpha = calAlpha(sideSpin, backSpin);

for(int i=0; i<100; i++)
{
            if(i>0)
              {
  if (height <0)
  {
   vA = 0.0;
   vB = 0.0;
   vC = 0.0;
  }
  else
  {
   vA = calVa(vA, vB, vC, v, alpha);
   vB = calVb(vA, vB, v);
   vC = calVc(vA, vC, v, alpha);
  }
}
```

```
private static double calV(double VA, double VB, double VC)
{
return(Math.sqrt(VA*VA+VB*VB+VC*VC));
}

private static double calAlpha(double SideSpin, double BackSpin)
{
return(Math.atan(SideSpin/BackSpin));
}

private static double calVa(double VA, double VB, double VC, double V,
double Alpha)
{
return((VA-0.0019*V*(0.25*VA-0.2*(VC*Alpha-VB))));
}

private static double calVb(double VA, double VB, double V)
{
return((VB-0.98-0.0019*V*(0.25*VB-0.2*VA)));
}

private static double calVc(double VA,  double VC, double V, double
Alpha)
{
return((VC-0.0019*V*(0.25*VC+0.2*VA*Alpha)));
}
```

Figure 18: Sample section of application version 3 – ReadWriteFile.java

University Of

4.9.2 Summary of the application version 3

The outcome of the application was successful. The VRML displayed a smooth curve trajectory. The final outcomes are shown below:

Table 3: Outcome results version 3

Expected result	Actual result
Two extra text fields will be implemented into the interface. In total, there are five text fields including: Speed (m/s), Vertical angle (deg), Right/left (deg), Blackspin (revs/sec) and Sidespin (revs/sec)	Five text fields are displayed which are: Speed (m/s), Vertical angle (deg), Right/left (deg), Blackspin (revs/sec) and Sidespin (revs/sec)
The additional formulas will be implemented into the Java application	Calculations are in place in the Java application
A new .wrl file to be generated (.wrl is extension of VRML)	New VRML file called 'golf.wrl' has been generated
The outcome effect should be a smooth curve of the golf ball trajectory	Outcome effect displayed a smooth curve of the golf ball trajectory

Screen shots of the application version 3 are placed in Appendix B.

University Of Hertfordshire

4.10 3D golf course 'VRML'

As shown above, version 3 application was a successful implementation, the next stage is to construct the 3D VRML golf course. The layout design of the golf course is an imitation of an existence web golf course. [10]

This section includes: Software used, Implementation of golf course and Screen shots of final VRML golf course.

4.11 Software

There are three applications used to code the 3D golf course as follows:

4.11.1 VrmlPad

VrmlPad is a powerful text editor for VRML programming. It offers visual support for the scene tree and document resource operations

The main key features are:

- Smart auto complete
- Dynamic errors detection,
- Syntax highlighting,
- Visual support of the Scene Tree
- Operation on the resources
- Integrated script debugger
- Automation and scripting
- Multiple documents
- Preview scenes

4.11.2 AC3D

AC3D is used to create 3D models for virtual-reality/simulation software such as: scientific, medical, and general data visualizations also rapid prototyping of 3D designs and high-resolution 3D renderings. AC3D is a relatively easy program to use which supports many 3D file formats including:

- POV-Ray
- VRML

University Of Hertfordshire

- RenderMan
- 3D Studio
- Lightwave
- MD2
- Quake BSP
- DirectX and many more.

4.11.3 NotePad

NotePad is a simple text-editor program, which contains many file format extensions, such as: html, xml, asp, jsp, wrl, java, php, xsl and many more.

4.12 Implementation of VRML golf course

In this implementation, the outcome should contain the following: interactive touch sensor on a golf ball, smooth navigation, a refresh button inside the scene, a realistic 3D golf course and a predefined coordinate of golf ball trajectory *(in VRML term, the 'KayValues' coordinate points)*

4.12.1 Sample section of the 3D golf course

The syntax below is used to define an object 'short grass', which can be re-used through out the VRML scene without re-enter of the actual code.

```
1 #VRML V2.0 utf8
2 PROTO shortgrass
3 [
4 exposedField SFString _type_name_ "Plane01"
5 exposedField SFNode _object_site_ NULL
6 exposedField SFInt32 _runtime_mode_ 0
7 exposedField SFNode texture ImageTexture
8 {
9 url "images/GreenGrass02.jpg"
10 }
11 field SFFloat _version_ 1
12 ]
13 {
```

Figure 19: Sample section of VRML golf course part 1

Line no.	Description
1	VRML header must be written in the first line of the file to be declared as a VRML file.
2	PROTO definition specifies the name for the new node type
4	SFString:list of characters enclosed within quotation marks.Typically used to specify the name of choice.
5	exposedField defines an interface with exposed field with a data type, an interface exposed-field name, and default value.
6	SFInt32:32-bit integer values. Large or small, positive or negative values without a decimal point.
7	- SFNode:Typically used to indicate a property node that controls how shape node will draw a shape(VRMLnode) - ImageTexture node specifies texture-mapping attributes
9	locate where image is stored
11	SFFloat:Floating-point values. Large or small, positive or negative values without a decimal point.

University Of Hertfordshire

```
14 DEF _sa_exported_object_ Group
15 {
16 children
17 [
18 DEF Plane01-FACES Shape
19 {
20 appearance Appearance
21 {
22 material Material
23 {
24 ambientIntensity 0.1
25 diffuseColor 0.5 0.5 0.5
26 shininess 0.2875
27 specularColor 0.045 0.045 0.045
28 }
```

Figure 20: Sample section of VRML golf course part 2

Line no.	Description
14	Group node together
16	Children field specifies a list of child nodes to be included in the group
18	DEF defines node and can be reused
20	specifies appearance attributes
22	specifies material attributes, by default it creates shaded white shapes
24	ambientIntensity controls how the material is affected by the ambient-light level of the world
25	diffuseColor field sets the red, green and blue-light
26	shininess controls the shininess of the material. higher values= more shiny
27	specifies the RGB specularColor of light reflected off a shape by spectacular reflections

```
29 texture IS texture
30 }
31 geometry IndexedFaceSet
32 {
33 coord Coordinate
34 {
35 point
36 [
37 -268.1 0 272.7, -268.1 0 0, 0 0 0,
38 0 0 272.7, 268.1 0 0, 268.1 0 272.7,
39 -268.1 0 -272.7, 0 0 -272.7, 268.1 0 -272.7
40 ]
41 }
42 texCoord TextureCoordinate
43 {
44 point
45 [
46 -36.61 -8.748, -22.36 14.51, 0.5 0.5,
47 -13.75 -22.75, 23.36 -13.51, 9.107 -36.76,
48 -8.107 37.76, 14.75 23.75, 37.61 9.748
49 ]
50 }
51 ccw FALSE
52 coordIndex
53 [
54 0, 1, 2, -1,    0, 2, 3, -1,       3, 2, 4, -1,
55 3, 4, 5, -1,    1, 6, 7, -1,       1, 7, 2, -1,
56 2, 7, 8, -1,    2, 8, 4, -1
57 ]
58 }
59 }
60 ]
61 }
62 }
```

Figure 21: Sample section of VRML golf course part 3

Line no.	Description
31	IndexedFaceSet creates face geometry and can be used as value for geometry field of a shape node Geometry defines the object shape
33	specifies a node listing coordinates available for building faces within face set
35	specifies a list of 3D coordinates used as geometry coordinates for a shape
42	TextureCoordinate specifies a list of texture coordinates
51	indicating faces in face set an unknown order
52	Specifies list of coordinate indexes describing the path of one or more polylines

University Of Hertfordshire

4.12.2 Summary of VRML golf course

The overall result was successful, with fully integrated interactive scene, smooth navigation, interactive touch sensor on a golf ball, a refresh button inside the scene, a predetermined coordinate of golf ball trajectory and a realistic 3D golf course.

Also some extra features were added which included multiple view points and a predefine URL's link to, golftek.com, p3proswing.com and lindsayputters.com (Mr. N Lindsay) *Screen shots of multiple view points are placed in Appendix E.*

4.13 Screen shots of final VRML golf course

The screen shots below showed the development stages in construction of the golf course:

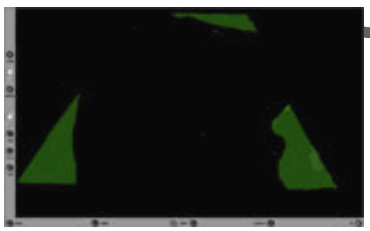


Figure 22: Stage 1

Figure 23: Stage 2

Figure 24: Stage 3

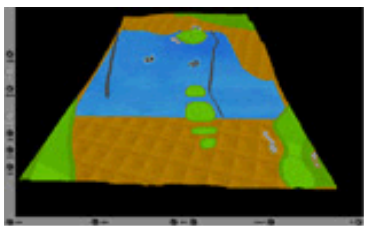


Figure 25: Stage 4

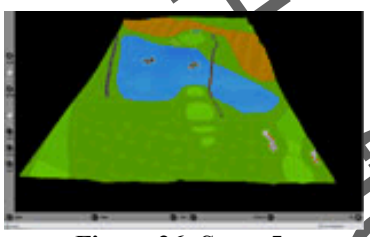


Figure 26: Stage 5

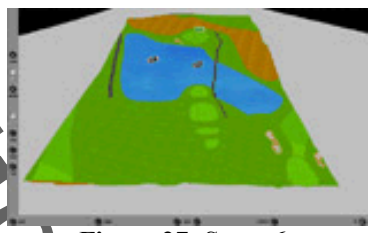


Figure 27: Stage 6

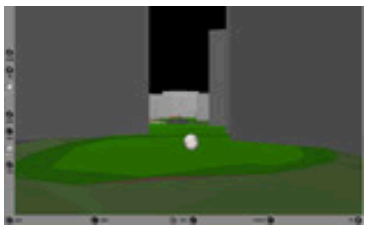


Figure 28: Stage 7

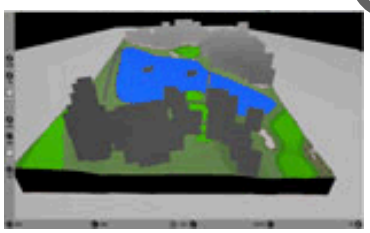


Figure 29: Stage 8

Figure 30: Stage 9

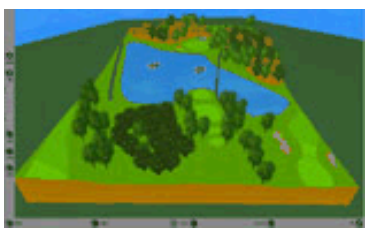


Figure 31: Stage 10

Figure 32: Stage 10

University of Hertfordshire

Figure 33: Stage 12

4.14 Final implementation

As understood from application version 1-3, the VRML source code is imbedded in the Java application. Under such situation, it would provide an unprofessional style in programming. To achieve a professional style, the application should be separated into four files including:

1. The Java interface
2. The application, which is the Java file contains the trajectory formulas.
3. The VRML golf course without the KeyValue coordinate points on the golf ball.
4. The generated Golf.wrl with the new KeyValue from end-users' inputs.

The expected outcome of the final application should be as follows:

- Four separated files:
    - Java interface with five text fields
    - Java application contains the trajectory formulas
    - Edited version of VRML golf course without the predefined coordinate on golf ball *(in VRML term 'KeyVaule')*, this will be replaced with the input data from end user
    - A new VRML file called 'Golf.wrl' should be generated with the input data from end user

University Of Hertfordshire

- The Golf.wrl should display the simulation of the golf ball trajectory.
- The trajectory of the golf ball can be changed at anytime with a different set of input data, inputted from the Java interface and submitted the data.
- To display the result again, simply click on the image with the description "Click me to replay you shot" in the golf course to see the new result.

4.14.1 Separation of files methods

Before separating the application into different files, there is a need to find out what are the methods to perform the task first. Below are the methods used to separate the file for the application.

BufferWriter – Write text to a character-output stream, buffering characters so as to provide for the efficient writing of single characters, arrays, and strings. [11]

System.out.print – method used to display values or variables on screen, but does not do a line feed at the end. If two or more items appear in a continuous line, use a succession of prints, followed by a println. [12]

StringBuffer – A string Buffer is like a string, but can be modified at any point in time, it contains some particular sequence of characters, but length and contain of the sequence can be changed through certain method calls. [13]

FileWriter: is used for output of character data to a disk file. Here is where it is in the hierarchy of Writer classes: [14]

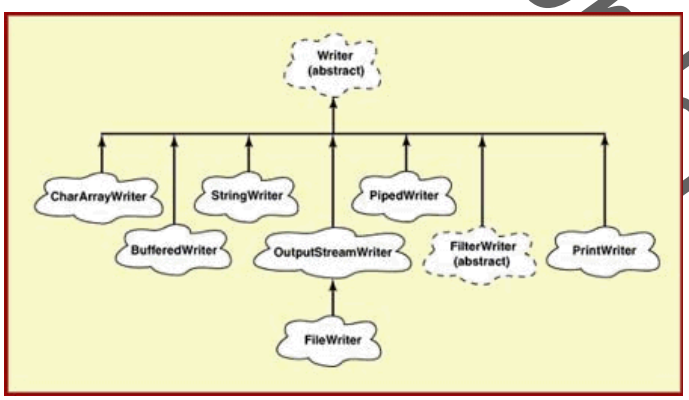


Figure 34: FileWriter hierarchy [14]

University Of Hertfordshire

4.14.2 Sample section of Final application

This is a sample section source code of the Final application.

```
private String getContents() {
  //...checks on Golftest are omitted elided
  StringBuffer contents = new StringBuffer();

  //declared here only to make visible to finally clause
  BufferedReader input = null;
  try {
   //use buffering
   //this implementation reads one line at a time
   //FileReader always assumes default encoding is OK!
   if(golfCourseChoice == 0)
      input = new BufferedReader( new
FileReader("C:\\FYP\\version5\\golf1_V13.wrl"));
    else if(golfCourseChoice == 1)
      input = new BufferedReader( new
FileReader("C:\\FYP\\version5\\golf2_V09.wrl"));
    else if(golfCourseChoice == 2)
      input = new BufferedReader( new
FileReader("C:\\FYP\\version5\\golf3_V04.wrl"));

    String line = null; //not declared within while loop
    while (( line = input.readLine()) != null){
     contents.append(line);
     contents.append(System.getProperty("line.separator"));
    }
```

Locate the VRML source file *(without 'KeyValues')*

Read the original VRML file, until it reaches the line "REPLACECODEHERE". This is where the calculated data from the trajectory formulas will be placed.

```
orignalFileContent = getContents();
      newContent = replace(orignalFileContent, "REPLACECODEHERE",
sb.toString());
```

Figure 35: Sample section of application version 3 – ReadWriteFile.java

The source codes for the final application is placed in *Appendix C*.

4.14.3 Flow Chart Final application

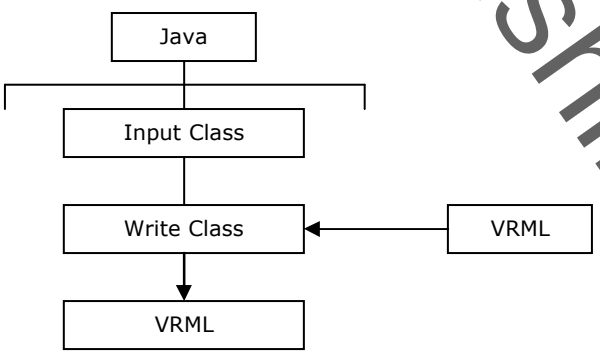


Figure 36: Final Application

University of Hertfordshire

4.14.4 Summary of the Final implementation

The outcome of the application/simulation was successful with bonus feature.

The results are shown in the table below:

Table 4: Outcome results Final application

Expected result	Actual result
Three separated files: Java interface, Java application and Edited version of VRML	Three files had been separated into Java interface, Java application and Edited version of VRML
A VRML file called 'Golf.wrl' should be generated	A 'Golf.wrl' had been generated
The Golf.wrl should display the simulation of the golf ball trajectory.	Golf.wrl displayed the golf ball trajectory using the values inputted from the interface
Values changed on the interface, in the VRML golf course click on the icon with "Click me to replay your shot" to view the latest trajectory	The "Click me to replay your shot" icon, displayed the latest trajectory.

The **Bonus** features are: Two extra golf courses had been added to the final application, in total there are three golf courses to choose from. The bonus golf courses provide an exciting experience using the simulation with various options to view the trajectory of a golf ball. Screens of the new golf courses are placed in *Appendix D.*

By adding extra golf courses, the application has proven to be very powerful. Not only can it calculate a complicated physics, it also displays a realistic and accurate trajectory of a golf ball in a 3D environment. As a result, the application has become more intelligent although it looks simple.

4.15 Summary

In this chapter, description are made for each stage of the design and implementation through out the process if this project. It will also cover the difficulties encountered during development of the application with summaries of the result for each stage.

University Of Hertfordshire

5. Testing

5.1 Introduction

Testing plays an important role in the development of the application. It is constantly being tested and debugged during the process as shown in various stages of development and implementation. In the final application, the test results for each golf course are displayed in three separate tables below:

5.2 Test result: Application and Golf course 01

Table 5: Test result: Application and Golf course 01

Test no.	Description	Expected result	Actual result	✓/✗
1	Interface: Insert values in all text fields on 'Golf course 01'	Result submitted	Result submitted	✓
2	Interface: Insert values on only some of the text fields	Result submitted	Result failed to submit	✗
3	Interface: Insert symbols or alphabets values in the text fields	Result failed to submit	Result failed to submit	✓
4	Interface: Insert any values, positive, negative and decimal points in all text fields	Result submitted	Result submitted	✓
5	Interface: Insert values '0' on all text fields	Result submitted	Result submitted	✓
6	Test to see if all values inserted, would 'Golf.wrl' be generated and viewable	'Golf.wrl' generated and viewable	'Golf.wrl' generated and viewable	✓
7	Test to see if some values inserted would 'Golf.wrl' be generated and viewable	'Golf.wrl' failed to generate or viewable	'Golf.wrl' failed to generate or viewable	✓
8	Test to see if symbols or alphabets is inserted, would 'Golf.wrl' be generated and viewable	'Golf.wrl' failed to be generated or viewable	'Golf.wrl' failed to be generated or viewable	✓
9	Test to see if any positive, negative and decimal points values inserted, would 'Golf.wrl' be generated and viewable	'Golf.wrl' generated and viewable	'Golf.wrl' generated and viewable	✓
10	Test to see if values '0'inserted would 'Golf.wrl' be generated and viewable	'Golf.wrl' generated and viewable	'Golf.wrl' is generated but failed to open	✗
11	Golf.wrl: click on 'golfTek' billboard to launch its web sites in a new window	'golfTek' web site launched in a new window	'golfTek' web site launched in a new window	✓
12	Golf.wrl: click on 'lindsayGolf' billboard to launch its web sites	'lindsayGolf' web site launched in a	'lindsayGolf' web site launched in a	✓

University Of Hertfordshire

	in a new window	new window	new window	
13	Golf.wrl: click on 'p3proswing' billboard to launch its web sites in a new window	'p3proswing' web site launched in a new window	'p3proswing' web site launched in a new window	✓
14	Golf.wrl: test to see if 'Click me to replay your shot' icon replays the simulation	icon replayed the simulation	icon replayed the simulation	✓
15	Golf.wrl: test to see if all VRML browser functions interact with the simulation the way it was designed	All VRML browser functions interacted with the simulation as planned	All VRML browser functions interacted with the simulation as planned	✓
16	Golf.wrl: test to see if the 7 predefined viewpoints displayed	The 7 predefined viewpoints displayed	The 7 predefined viewpoints displayed	✓
17	Golf.wrl: test to see if resize web browser, would golf course automatically adjust to the browser size	golf course automatically adjusted to the browser size	golf course automatically adjusted to the browser size	✓
18	View Golf.wrl in FireFox with ParalleGraphics Cortona plug-in	Golf.wrl opened	Golf.wrl opened	✓

5.3 Test result: Application and Golf course 02

Table 6: Test result: Application and Golf course 02

Test no.	Description	Expected result	Actual result	✓/✗
1	Interface: Insert values in all text fields on 'Golf course 02'	Result submitted	Result submitted	✓
2	Interface: Insert values on only some of the text fields	Result submitted	Result failed to submit	✗
3	Interface: Insert symbols or alphabets values in the text fields	Result failed to submit	Result failed to submit	✓
4	Interface: Insert any values, positive, negative and decimal points in all text fields	Result submitted	Result submitted	✓
5	Interface: Insert values '0' on all text fields	Result submitted	Result submitted	✓
6	Test to see if all values inserted, would 'Golf.wrl' be generated and viewable	'Golf.wrl' generated and viewable	'Golf.wrl' generated and viewable	✓
7	Test to see if some values inserted would 'Golf.wrl' be generated and viewable	'Golf.wrl' failed to generate or viewable	'Golf.wrl' failed to generate or viewable	✓
8	Test to see if symbols or alphabets is inserted, would 'Golf.wrl' be generated and viewable	'Golf.wrl' failed to be generated or viewable	'Golf.wrl' failed to be generated or viewable	✓
9	Test to see if any positive, negative and decimal points	'Golf.wrl' generated and	'Golf.wrl' generated and	✓

University Of Hertfordshire

	values inserted, would 'Golf.wrl' be generated and viewable	viewable	viewable	
10	Test to see if values '0'inserted would 'Golf.wrl' be generated and viewable	'Golf.wrl' generated and viewable	'Golf.wrl' was generated but failed to open	✗
11	Golf.wrl: test to see if 'Click here to replay your shot' sign pole replays the simulation	Sign pole replayed the simulation	Sign pole icon replayed the simulation	✓
12	Golf.wrl: test to see if all VRML browser functions interact with the simulation the way it was designed	All VRML browser functions interacted with the simulation as planned	All VRML browser functions interacted with the simulation as planned	✓
13	Golf.wrl: test to see if resize web browser, would golf course automatically adjusts to the browser size	golf course automatically adjusted to the browser size	golf course automatically adjusted to the browser size	✓
14	View Golf.wrl in FireFox with ParalleGraphics Cortona plug-in	Golf.wrl opened	Golf.wrl opened	✓

5.4 Test result: Application and Golf course 03

Table 7: Test result: Application and Golf course 03

Test no.	Description	Expected result	Actual result	✓/✗
1	Interface: Insert values in all text fields on 'Golf course 03'	Result submitted	Result submitted	✓
2	Interface: Insert values on only some of the text fields	Result submitted	Result failed to submit	✗
3	Interface: Insert symbols or alphabets values in the text fields	Result failed to submit	Result failed to submit	✓
4	Interface: Insert any values, positive, negative and decimal points in all text fields	Result submitted	Result submitted	✓
5	Interface: Insert values '0' on all text fields	Result submitted	Result submitted	✓
6	Test to see if all values inserted, would 'Golf.wrl' be generated and viewable	'Golf.wrl' generated and viewable	'Golf.wrl' generated and viewable	✓
7	Test to see if some values inserted would 'Golf.wrl' be generated and viewable	'Golf.wrl' failed to generate or viewable	'Golf.wrl' failed to generate or viewable	✓
8	Test to see if symbols or alphabets are inserted, would 'Golf.wrl' be generated and viewable	'Golf.wrl' failed to be generated or viewable	'Golf.wrl' failed to be generated or viewable	✓
9	Test to see if any positive, negative and decimal points values inserted, would 'Golf.wrl'	'Golf.wrl' generated and viewable	'Golf.wrl' generated and viewable	✓

University Of Hertfordshire

	be generated and viewable			
10	Test to see if values '0'inserted would 'Golf.wrl' be generated and viewable	'Golf.wrl' generated and viewable	'Golf.wrl' was generated but failed to open	✗
11	Golf.wrl: click on 'golfTek' billboard to launch its web sites in a new window	'golfTek' web site launched in a new window	'golfTek' web site launched in a new window	✓
12	Golf.wrl: click on 'lindsayGolf' billboard to launch its web sites in a new window	'lindsayGolf' web site launched in a new window	'lindsayGolf' web site launched in a new window	✓
13	Golf.wrl: test to see if 'Click golf bag to replay your shot' replays the simulation	Golf bag replayed the simulation	Golf bag replayed the simulation	✓
14	Golf.wrl: test to see if all VRML browser functions interact with the simulation the way it was designed	All VRML browser functions interacted with the simulation as planned	All VRML browser functions interacted with the simulation as planned	✓
15	Golf.wrl: test to see if the 7 predefined viewpoints display	The 7 predefined viewpoints displayed	The 7 predefined viewpoints displayed	✓
16	Golf.wrl: test to see if resize web browser, would golf course automatically adjusts to the browser size	golf course automatically adjusted to the browser size	golf course automatically adjusted to the browser size	✓
17	View Golf.wrl in FireFox with ParalleGraphics Cortona plug-in	Golf.wrl opened	Golf.wrl opened	✓

5.5 Summary

Overall the test results are very ideal as most of the results are found out to be positive except two minor issues.

Issue 1: In the interface when values are not inserted in all text fields before submission, the application can not receive the input data as the text field is empty.

Fix 1: What should appear on the interface should be an instruction putting on top "Please insert values into ALL text fields in order to proceed forward"

Issue 2: In the interface when value '0' is inserted in the text field, the result is submitted but it failed to launch the Golf.wrl file. This is because there is no value to be calculated and inserted into 'KeyValue'

University Of Hertfordshire

Fix 2: Create another instruction to warn users that entering value '0' the simulation will not be displayed. In general the simulation is built to display a golf ball trajectory, therefore if end- users decided to enter '0' as the value, the golf ball will go nowhere as a result.

6. Conclusion

Doing this project has taught the author a lot on 3-dimensional environment especially working with very powerful languages to create a highly constructed simulation to produce an accurate trajectory of a golf ball.

The final implemented application has proven that it achieved all the aims and objectives stated in *chapter 1*. It even exceeded the specifications set on the project with extra golf courses added to the application bringing more attraction in using the simulation. Also, it proved that the application is flexible in which modification can be easily done to put in new golf courses.

There is one problem on the simulation, occasionally the golf ball went underground which was caused by mis-calculation in the formats provided. At the end of the calculation the values should return to 0,0,0 which was the origin but in this case the calculation produced negative values, leading to the golf ball returning below ground level.

Also to improve a better landing of the golf ball, the coordinates of the 3-dimensional foundation can be transferred into a program/ languages which can map out the 3-dimensional ground and can reset the origin to of the course, let say 0.5 to avoid collision between the golf ball and the ground.

Overall, the project was very challenging and at times it was almost impossible to move on, as each component needed to link with each other in order to proceed forward. Although the project was very challenging, it was interesting as well, which gave the motivation to resolve the issues and pushed on as quickly as possible to see the final outcome.

University Of Hertfordshire

With the help of feedbacks obtained from the questionnaires, further improvement can be made as follows:

- Add a company logo on the interface as a background to give a professional image also branding the application.
- Add detailed instruction on the interface, on how to use the application and what type of symbol can it support.
- Create more golf courses to choose from
- The simulation can be transferred online upon request, as VRML runs on the web browser. This could be either for advertisement of Mr Lindsay's future golf driving range or a web base demonstration of teaching golfer how to improve their skill.

University Of Hertfordshire

REFERENCES

[1] H.M.Deitel and P.J.Deitel, **Java How To Program 5th Edition,** Prentice Hall, United States of America, 2003

[2] http://www.web3d.org/x3d/specifications/vrml/VRML1.0/index.html

The Virtual Reality Modelling Language

[3] http://www.web3d.org/x3d/overview.html

X3D Overview

[4] http://www.sitepoint.com/article/java-language-basics

Java Language Basics ***By Kevin Yank***

[5] H.M.Deitel and P.J.Deitel, **Java How To Program 5th Edition,** Prentice Hall, United States of America, 2003

[6] http://en.wikipedia.org/wiki/Java_3D

Java 3D

[7] http://www.javaworld.com/javaworld/jw-12-1998/jw-12-media.html

3D graphics programming in Java, Part 1: Java 3D

[8] http://www.thedesignshop.com/history.htm

Golf ball and golf history

[9] Daniel Selman, **Java 3D Programming,** Manning, 2003

[10] http://mathworld.wolfram.com/EulerForwardMethod.html

By Eric W. Weisstein, **The Forward Euler Method**, 1999-2006

[11] Microsoft Excel Help 2003. ATAN

[12] http://www.parallelgraphics.com

[13] http://java.sun.com/j2se/1.4.2/docs/api/java/io/BufferedWriter.html

Sun Microsystems

[14] Peter McBride, **Java Made SimpleB**, Made Simple Books,Great Britain,1997-99

[15] http://java.sun.com/j2se/1.4.2/docs/api/java/lang/StringBuffer.html

Sun Microsystems

[16] http://chortle.ccsu.ctstateu.edu/CS151/Notes/chap83/ch83_3.html

By Bradley Kjell, Central Connecticut State University

[17] http://www.web3d.org/x3d/specifications/ISO-IEC-19776-X3DEncodings-XML-ClassicVRML/Part02/examples.html

Extensible 3D (X3D) encodings Part 2: Classic VRML encoding

University Of Hertfordshire

BIBLIOGRAPHY

Rodger Lea, Kouichi Matsuda and Ken Miyashita, **Java for 3D and VRML worlds**, New Riders, United States of America, 1996

Aaron E. Walsh and Doug Gehringer, **Java 3D API Jump Start,** Prentice Hall PRT, United States of America, 2002

H.M.Deitel and P.J.Deitel, **Java How To Program 5th Edition,** Prentice Hall, United States of America, 2003

Peter McBride, **Java Made SimpleB**, Made Simple Books, Great Britain, 1997-99

Andrea L. Ames, David R. Nadeau and John L. MoreLand, **VRML2.0 Source Book**, Wiley, United States of America, 1997

Useful Site For X3d

http://www.web3d.org/
http://xml.coverpages.org/vrml-X3D.html
http://www.realism.com/Web3D/x3d/tutorials.html
http://www.xj3d.org/tutorials/general_sai.html
http://www.xj3d.org/javadoc/index.html
http://www.xj3d.org/tutorials/examples/sai/SimpleSAIDemo.java
http://www.xj3d.org/tutorials/examples/sai/FieldAccessDemo.java
http://www.xj3d.org/requirements.html
https://j3d-core.dev.java.net/servlets/ProjectDocumentList?folderID=2963&expandFolder=2963&folderID=0
https://games-binaries.dev.java.net/build/index.html
http://www.mozilla.org/rhino/download.html
http://www.vrmlsite.com/#vrmljava
http://www.vrmlsite.com/feb97/a.cgi/spot2.html
http://www.web3d.org/x3d/specifications/ISO-IEC-19777-1-X3DLanguageBindings-ECMAScript/Part1/examples.html

University Of Hertfordshire

X3D and XML

http://www.xml.com/pub/a/2003/08/06/x3d.html?page=2#x3dBrowsers

X3D Example

http://www.mediamachines.com/samplefiles.html

http://www.bitmanagement.de/

http://www.bitmanagement.de/demos/customer_apps.en.html#1

X3D

http://www.web3d.org/x3d/overview.html

Information on VRML

http://philliphansel.com/tutorials/tutorials.htm

http://www.vrmlsite.com/

http://tecfa.unige.ch/guides/vrml/vrmlman/node5.html Proto

http://h-anim.org/Specifications/H-Anim1.0/ (eg of SFstring)

http://www.parallelgraphics.com/developer/products/vrmlpad/reference/proto/
(library proto)

http://accad.osu.edu/~pgerstma/protolib/index.html?node=VisibleTransformsPackage
(refer proto)

VRML Script

http://docs.rinet.ru/WebPub/ch41.htm (How to use VRML script)

http://developer.blaxxun.com/developer/contact/3d/spec/vrmlscript/vrmlscript.html
(Proposal 4VRMLScript Node Authoring Interface)

http://zing.ncsl.nist.gov/~gseidman/vrml/repos/refer.txt (examples of VRMLScript)

Xvrml Included Right Handed Rule

http://www.xvrml.net/tutorial/instruction.php?p=3dconcept

Java 3d And Vrml 97

http://www.javaworld.com/javaworld/jw-01-1999/jw-01-media.html

University Of Hertfordshire

VRML1.0

http://www.web3d.org/x3d/specifications/vrml/VRML1.0/index.html

VRML Collision Node

http://www.lighthouse3d.com/vrml/tutorial/index.shtml?

VRML Eg:Collision Node

http://www.vrmlworlds.com/developer/tutorials/basic/collision.html

Object-To-Object Collision Detection Interface

http://www.parallelgraphics.com/developer/products/cortona/extensions/collision

VRML History

http://vrmlworks.crispen.org/history.html

Good VRML History

http://www-winfo.uni-siegen.de/vrmlHistory/docs/partVH/introduction.html

Vrml1 Vs Vrml2

http://www.engr.mun.ca/~adfisher/vrml/sld003.htm

VRML Cortona Client

http://www.parallelgraphics.com/products/cortona/

VRML Web3d

http://www.web3d.org/x3d/vrml/

Vrml97 Web3d

http://www.web3d.org/x3d/specifications/vrml/

VRML 1.0

http://www.web3d.org/x3d/specifications/vrml/VRML1.0/index.html

VRML Browsers

http://www.web3d.org/x3d/vrml/tools/viewers_and_browsers/

University Of Hertfordshire

VRML 97 Spec

http://tecfa.unige.ch/guides/vrml/vrml97/spec/

VRML 2.0 Vs X3D

http://www.web3d.org/x3d/x3d_vs_vrml.html

VRML SYATAX

http://www.agocg.ac.uk/train/vrml2rep/part1/guide3.htm

MICROSOFT EXCEL

If Statement In Excel

http://pubs.logicalexpressions.com/Pub0009/LPMArticle.asp?ID=225

http://personal-computer-tutor.com/IFstatement.htm

Atan Excel

http://www.greytrout.com/manuals/SS_user_guide/node43.html

http://www.techonthenet.com/excel/formulas/atan.php

Sqrt Excel

http://www.techonthenet.com/excel/formulas/sqrt.php

Maths_Euler's Method Golf Ball

http://www.math.ubc.ca/~feldman/demos/demo1.html

http://www.ugrad.math.ubc.ca/coursedoc/math100/notes/mordifeqs/euler.html

http://www.math.montana.edu/frankw/ccp/calculus/des/euler-pictures/learn.htm

http://www.math.montana.edu/frankw/ccp/calculus/des/euler-formulas/body.htm

http://director-online.com/buildArticle.php?id=346

http://www.rose-hulman.edu/~millerts/Classes/code%20descripition.html

http://mathworld.wolfram.com/EulerForwardMethod.html

Golf Information

Dimples Vs Smooth Golf Ball

http://wings.avkids.com/Book/Sports/instructor/golf-01.html

University Of Hertfordshire

History of golf ball

http://www.thedesignshop.com/history.htm

http://library.thinkquest.org/10556/english/high/history/hist05.htm

Brief history of golf

http://www.tenerifegolfholidays.co.uk/About%20Golf.htm

Find The Right Ball For Your Game

http://www.learnaboutgolf.com/equipment/balls.html

Golf Ball

http://ffden-2.phys.uaf.edu/211_fall2002.web.dir/josh_fritts/ball.html

Golf

http://searchwarp.com/swa42729.htm

http://www.napit.co.uk/viewus/infobank/golf/index.php(history of golf)

http://www.tourcanada.com/golfhist.htm(history of golf)

Golf Course

http://www.martindalecenter.com/Calculators1C_1_Sport.html

http://www.ecs.syr.edu/centers/simfluid/red/golf.html

http://www.ecs.syr.edu/centers/simfluid/red/golf_exp.html

http://www.ecs.syr.edu/centers/simfluid/red/golf_top.html

http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?tp=&arnumber=144825&isnumber=3878

http://www.fxgolf.com/OTHER/fxgolftips.asp

http://www.fxgolf.com/Screenshot/screenshot.asp

http://www.horsepowergolf.com/TourVanSchedule.asp

http://www.freepatentsonline.com/5390927.html

http://www.franklygolf.com/Speak/aerodynamics.asp

http://www.franklygolf.com/Speak/QA_ball_test.asp

http://sun4a.vaniercollege.qc.ca/golf/Instruction.html

http://sun4a.vaniercollege.qc.ca/golf/index.html

http://www.tourcanada.com/golfhist.htm

University Of Hertfordshire

http://www.coursemaster.net/About.asp

http://www.coursemaster.net/ProductsNavigator.asp

http://www.coursemaster.net/Products3DGuides.asp

3D Golf Simulation

http://www.sports-coach.com/mainprod.aspx (3D golf simulation)

http://www.sports-coach.com/linksls3.aspx

http://www.topgolf.co.uk/chigwell.htm

http://www.golftek.com/

http://www.parmastersgolf.com/fw/main/Golf_Simulators-268.html

http://www.forgefx.com/demos/golf-0.17.htm

http://www.3dnature.com/golfarc.html(3D golf courses)

http://www.turbosquid.com/FullPreview/Index.cfm/ID/299793 (3d golf course)

http://www.superscape.com/games/title.php?CGT0_3D,overview (3D game)

http://www.kahootz.com.au/kz/template/home,ShowcaseExample6.vm(3D golf course)

JAVA RESEARCH

Java 3d Site

http://java.sun.com/products/java-media/3D/collateral/

http://www.javaworld.com/javaworld/jw-12-1998/jw-12-media.html

http://www.javaworld.com/javaworld/jw-01-1999/jw-01-media.html

http://www.anfyteam.com/panfy3d.html

http://www.sdsc.edu/~nadeau/Courses/Siggraph99/

http://www.freeprogrammingresources.com/java-graphics-tutorial.html

http://java.sun.com/developer/onlineTraining/java3d/

http://java.sun.com/developer/onlineTraining/javaai/jai/index.html

Basic Java

http://www.sitepoint.com/article/java-language-basics

Buffer Replace

http://www.javapractices.com/Topic80.cjp

University Of Hertfordshire

Read And Write A File

http://www.javapractices.com/Topic42.cjp

String Buffer

http://java.sun.com/j2se/1.4.2/docs/api/java/lang/StringBuffer.html

How to Use File Streams

http://java.sun.com/docs/books/tutorial/essential/io/filestreams.html

Read a file and print, using BufferedReader and System.out

http://www.java2s.com/ExampleCode/File-Input-Output/ReadafileandprintusingBufferedReaderandSystemout.htm

Reading Numbers from a Text File, Line by Line

http://www.java2s.com/ExampleCode/File-Input-Output/ReadingNumbersfromaTextFileLinebyLine.htm

3d World

http://www.openworlds.com/x3d.html

http://www.javaworld.com/javaworld/jw-01-1999/jw-01-media.html

Right Handed Rule

http://www.schorsch.com/kbase/glossary/right_hand_rule.html

http://www.schorsch.com/kbase/glossary/right_hand_rule.html

University Of Hertfordshire

APPENDICES

Appendix A

Example of 3D languages

Three different languages including Java 3D, VRML and X3D were tested to determine which language is best suited for this project.

Java 3D

This example is from 'Java 3D programming, By Daniel Selman' [8]

- Background geometry, in this case the scene is placed within a Sphere
- Textured geometry, an image is applied to the inside of the background Sphere to give the illusion of a distant skyline.
- Lighting ,a single directional light is created to provide depth cues through rendering.
- Geometry, a second smaller Sphere is placed within the scene.
- Appearance, the smaller Sphere has an Appearance and Material associated with it that interacts with directional light to produce a shaded, coloured effect.
- Animation, a PositionInterpolator behaviour is attached to the smaller Sphere to move it left and right using a complex time function (Alpha).

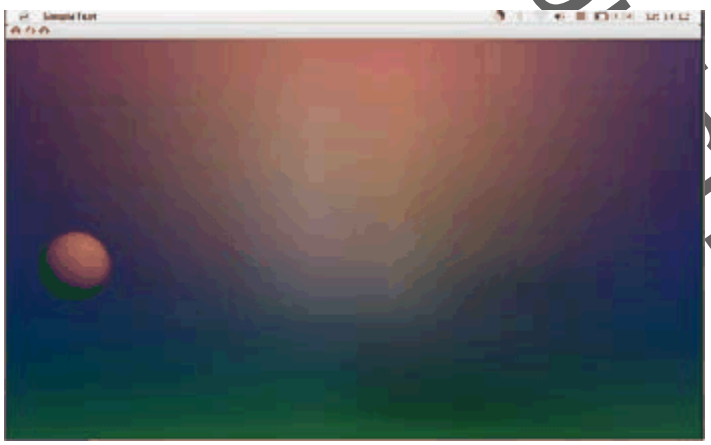


Figure 37: Java3D example from Mac OSX

University Of Hertfordshire

VRML 2.0

In this VRML program, it contains five sphere objects and a touch sensor.
When touch sensor is activated, the first sphere will display a predefined animation.
Activate touch sensor 2nd time, the first sphere will stop at current location and second sphere start the predefined animation and so on.

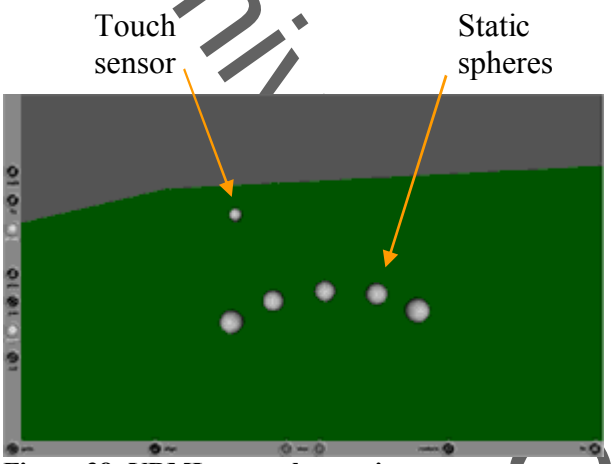


Figure 38: VRML example - static

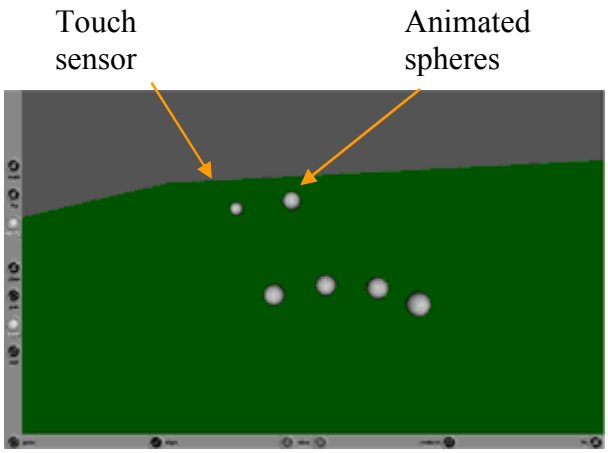


Figure 39: VRML example - Animated

X3D

X3D is the next generation of VRML 2.0, this example is from web3d.org [16] which shows an animated robot.

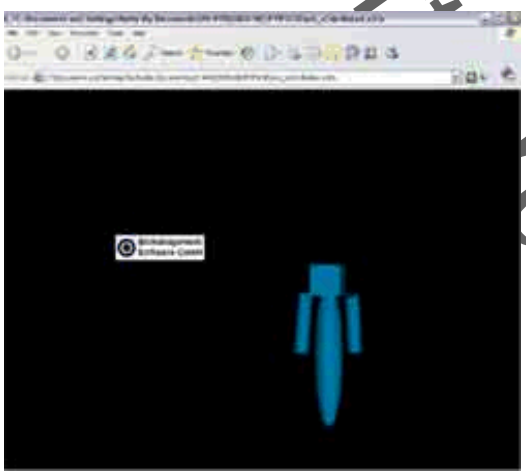


Figure 40: X3D – Animated robot

University Of Hertfordshire

X3D Source code

The source below shows that X3D is very similar to VRML.

```
#X3D V3.0 utf8
PROFILE Immersive
META "filename" "Robot.x3dv"

EXTERNPROTO Shuttle [
  initializeOnly SFTime  rate
  initializeOnly SFFloat distance
  initializeOnly MFNode  children
  inputOutput    SFTime  startTime
  inputOutput    SFTime  stopTime
  initializeOnly SFBool  loop
]
"ShuttlesAndPendulums.x3dv#Shuttle"

EXTERNPROTO Pendulum [
  initializeOnly SFTime  rate
  initializeOnly SFFloat maxAngle
  initializeOnly MFNode  children
  inputOutput    SFTime  startTime
  inputOutput    SFTime  stopTime
  initializeOnly SFBool  loop
]
"ShuttlesAndPendulums.x3dv#Pendulum"

Viewpoint { description "Watch Robot"
  position 0 0 150
}

DEF Near ProximitySensor { size 200 200 200 }

DEF Walk Shuttle { stopTime 1 rate 10 distance 20
  children [ # The Robot
    Transform { rotation 0 1 0 1.57
      children [
        Shape {
          appearance DEF A Appearance {
            material Material { diffuseColor 0 0.5 0.7 }
          }
          geometry Box { } # head
        }
        Transform { scale 1 5 1 translation 0 -5 0
          children Shape {    # body
            appearance USE A
            geometry Sphere { }
          }
}
        Transform { rotation 0 1 0 1.57 translation 1.5 0
0
          children DEF Arm Pendulum {
            stopTime 1 rate 1 maxAngle 0.52 # 30 degrees
            children [
              Transform { translation 0 -3 0
                children Shape {
                  appearance USE A
                  geometry Cylinder {  height 4 radius 0.5 }
                }
              }
            ]
          }
        }
        # duplicate arm on other side and flip so
        # it swings in opposition
        Transform { rotation 0 -1 0 1.57 translation -1.5 0
0
          children USE Arm
        }
      ]
    }
  ]
}
ROUTE Near.enterTime TO Walk.startTime
ROUTE Near.enterTime TO Arm.startTime
ROUTE Near.exitTime TO Walk.stopTime
ROUTE Near.exitTime TO Arm.stopTime
```

Figure 41: X3D example source code

rdshire

Appendix B

Implementation of application versions 1-3

Version 1 – Initial application

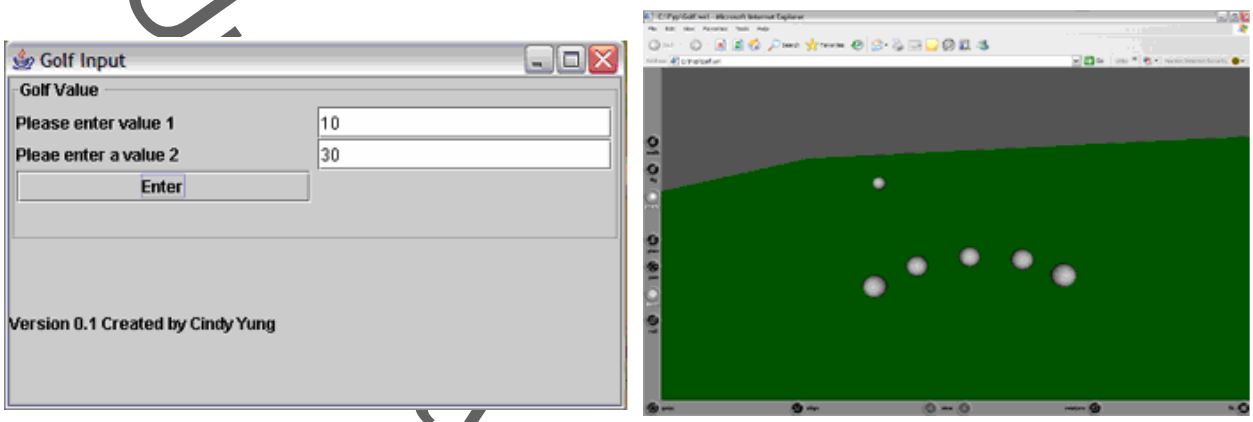


Figure 42: Initial application (Version 1)

Version 1 - GolfInput.java Source code

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.*;
/**
 * <p>Final Year Project Code
 * <p>Main class to the image processing class
 * give user choice to choose from either automatic
 * or manual processing mode
 *
 * @author Cindy Yung
 * @version 1.0
 */
public class GolfInput extends JFrame
{
  ////////////////////////// INSTANCE VARIABLES ///////////////////////////////
        private JButton plainButton,FancyButton,buttonEnter,buttonManual;
        private JPanel ControlPanel;
        private JPanel TopPanel;
        private JPanel BottomPanel;
        private JLabel valLabel, val2abel, Version;
        private JTextField val1, val2;
        private GridLayout grid1;
 /**
   * This constructor will perform the creatiion of the GUI
   * and linking to the other processing GUI
   *
   */
 public GolfInput()
 {
       super("Golf Input");
       //setting the GUI with windows feel
//        WindowUtilities.setNativeLookAndFeel();
       grid1 = new GridLayout(2,1,5,0);
       Container container = getContentPane();
       container.setLayout(grid1);
```

Ci

```
        ControlPanel = new JPanel();
        Border ControlPanelBorder = BorderFactory.createTitledBorder("Golf Value ");
        ControlPanel.setBorder(ControlPanelBorder);
        valLabel = new JLabel("Please enter value 1");
        ControlPanel.add(valLabel);

        ControlPanel.setLayout(new GridLayout(2,2,5,0));
        val1 = new JTextField(10);
        ControlPanel.add(val1);

        ControlPanel.setLayout(new GridLayout(3,3,5,0));
        val2abel = new JLabel("Pleae enter a value 2");
        ControlPanel.add(val2abel);

        ControlPanel.setLayout(new GridLayout(3,3,5,0));
        val2 = new JTextField(10);
        ControlPanel.add(val2);

        ControlPanel.setLayout(new GridLayout(4,4,5,0));
        buttonEnter = new JButton("Enter");
        buttonEnter.addActionListener(new ActionListener() {
          public void actionPerformed(ActionEvent ae) {
//            dispose();
            int text = Integer.parseInt(val1.getText());
            int text2 = Integer.parseInt(val2.getText());
            new ReadWriteFile(text, text2);
        }
        });
        ControlPanel.add(buttonEnter);

        BottomPanel = new JPanel();
        BottomPanel.setLayout(new GridLayout(1,1));

        Version = new JLabel("Version 0.1 Created by Cindy Yung");
        BottomPanel.add(Version);

       // container.add(TopPanel);
        container.add(ControlPanel);
        container.add(BottomPanel);

        setSize(450,270);
        Dimension screen = Toolkit.getDefaultToolkit().getScreenSize();
        int x = (screen.width - 450) / 2;
        int y = (screen.height - 270) / 2;
        setLocation(x, y);
        setVisible(true);
    try {
      jbInit();
    }
    catch(Exception e) {
      e.printStackTrace();
    }

 }
/** Main class of the program
   * create an instance of the ImageProcessingMain class
   */
 public static void main(String args[])
 {
        GolfInput application = new GolfInput();
        application.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
 }
  private void jbInit() throws Exception {
  }
}
```

Version 1 - ReadWriteFile.java Source code

```
import java.io.*;

public class ReadWriteFile {

  public ReadWriteFile(){
  }

  public ReadWriteFile(int val, int val2){
    setContent(val, val2);
  }

  static public void setContent(int Pos1, int Pos2)
  {
    System.out.println("New input ++++++++++++++++++ " + Pos1 );
    System.out.println("New input 2 +++++++++++++++++ " + Pos2 );
    try {
       //FileWriter = new FileWriter
       BufferedWriter out = new BufferedWriter(new FileWriter("c:\\Fyp\\Golf.wrl", false));

       out.write("#VRML V2.0 utf8\n\n" +
            "NavigationInfo { headlight TRUE type [\"EXAMINE\", \"ANY\"]}  \n" +
            "Background { skyColor .333 .333 .333 }  \n"+
            "Viewpoint {       \n  " +
            "   position 5.609 7.689 13.201\n   "+
            "   orientation -.647 .76 -0.068 .512 \n   "+
            "   description \"click!\"          \n"  +
            " }                       \n \n"+
            "   DEF ROOT Group {  \n"+
            "   children [    \n\n"+
            "   Transform {   \n"+
            "        children [                        \n"+
            "          DEF BOX Transform {             \n"+
            "            translation -4 0.5 0       \n"+
            "            children [           \n"+
            "                    Shape {          \n"+
            "                       appearance Appearance { material Material {} }    \n"+
            "                       geometry Sphere { radius 0.5}       \n"+
            "                    }                              \n"+
            "            ]                         \n"+
            "          }                        \n"+
            "       ]                       \n"+
            "     }                      \n"+
            "     Transform {          \n"+
            "       children [     \n"+
            "         Transform {                                    \n"+
            "           rotation 0 1 0 -.628                        \n"+
            "           children USE BOX                           \n"+
            "           }                                         \n"+
            "       ]                                        \n"+
            "     }                                        \n"+
            "                                            \n"+
            "     Transform {                             \n"+
            "       children [                       \n"+
            "         Transform {                   \n"+
            "           rotation 0 1 0 -1.256  \n"+
            "           children USE BOX              \n"+
            "         }                          \n"+
            "       ]                        \n"+
            "     }                       \n"+
            "                           \n"+
            "     Transform {          \n"+
            "       children [     \n"+
            "         Transform {                                    \n"+
            "           rotation 0 1 0 -1.885                       \n"+
            "           children USE BOX                           \n"+
            "         }                                         \n"+
```

```
          "       ]                                                    \n"+
          "     }                                                      \n"+
          "                                                            \n"+
          "     Transform {                                            \n"+
          "       children [                                           \n"+
          "         Transform {                                        \n"+
          "           rotation 0 1 0 -2.513                            \n"+
          "           children USE BOX                                 \n"+
          "         }                                                  \n"+
          "       ]                                                    \n"+
          "     }                                                      \n"+
          "                   Shape {          \n"+
          "                     appearance Appearance { material Material {diffuseColor 0.0 1.0
0.0} }      \n"+
          "                     geometry Box { size 100 0.1 100}                    \n"+
          "                   }                                                     \n"+
          "                                                                         \n"+
          "   ]                                                                     \n"+
          "}                                                                        \n"+
          "                                                                         \n"+
          "                                                                           \n"+
          "                                                                         \n"+
          "     Transform {                                                         \n"+
          "       translation -3 5 0                                                \n"+
          "        children [                                                       \n"+
          "          DEF TOUCH TouchSensor {}      \n"+
          "          Shape {                 \n"+
          "            appearance Appearance { material Material {} }                 \n"+
          "            geometry Sphere { radius .25}                                  \n"+
          "          }                                                                \n"+
          "        ]                                                                  \n"+
          "     }                                                                     \n"+
          "                                                                           \n"+
          "     DEF TIME TimeSensor {                                                 \n"+
          "       loop TRUE                                                         \n"+
          "     }                                                                   \n"+
          "                                                                       \n"+
          "     DEF PI PositionInterpolator {      \n"+
          "       key [ 0 , .5 1 ]           \n"+
          "       keyValue [ 0 0 0, 0 5 "+Pos1+", 0 0 "+Pos2+"]              \n"+
          "     }                                                                   \n"+
          "                                                                       \n"+
          "     DEF SCRIPT Script {                                                   \n"+
          "       eventIn SFBool set_val                                            \n"+
          "                                                                     \n"+
          "       field SFBool   first  TRUE       \n"+
          "       field SFInt32  i      0                                         \n"+
          "       field SFNode   timer  USE TIME                                    \n"+
          "       field SFNode   interp USE PI                                    \n"+
          "       field SFNode   root   USE ROOT                                  \n"+
          "                                   \n"+
          "       directOutput TRUE                \n"+
          "       url [\"javascript:                                              \n"+
          "                                                                     \n"+
          "         function set_val (clicked) {          \n"+
          "           if (clicked) {                \n"+
          "             if (first) {            \n"+
          "               first = false;   \n"+
          "             Browser.addRoute(timer, 'fraction_changed', interp, 'set_fraction'); \n"+
          "             }                                                         \n"+
          "             else {                                                                \n"+
          "               if (i > 0) { Browser.deleteRoute(interp, 'value_changed',
root.children[i-1], 'set_translation'); }  \n"+
          "               else       { Browser.deleteRoute(interp, 'value_changed',
```

```
root.children[root.children.length-1], 'set_translation'); }       \n"+
            "               } \n"+
            "                                                                                \n"+
            "               Browser.addRoute(interp, 'value_changed', root.children[i],
'set_translation');     \n"+
            "               i++;                                                \n"+
            "               i %= root.children.length;                          \n"+
            "             }                                                  \n"+
            "           }                                                  \n"+
            "                                                          \n"+
            "           \"]                                            \n"+
            "      }                                                \n"+
            "                                                   \n"+
            "      ROUTE TOUCH.isActive TO SCRIPT.set_val  \n");

       out.close();
    } catch (IOException e) {
      }
  }
}
```

Version 2 - application

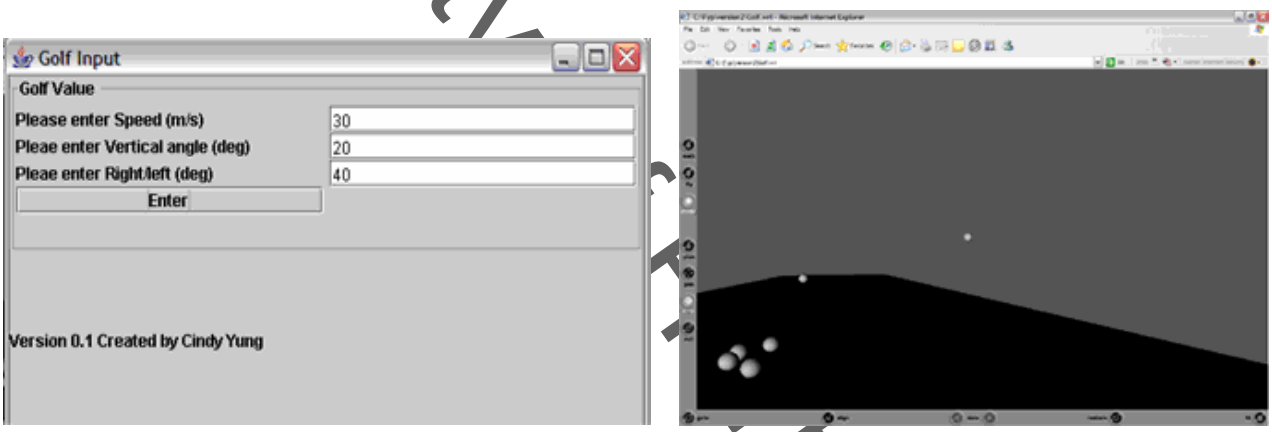


Figure 43: Application Version 2

University of Hertfordshire

Version 2 - GolfInput.java Source code

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.*;


/**
 * <p>Final Year Project Code
 * <p>Main class to the image processing class
 * give user choice to choose from either automatic
 * or manual processing mode
 *
 * @author Jeffrey Yung
 * @version 1.0
 */

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.*;

/**
 * <p>Final Year Project Code
 * <p>Main class to the image processing class
 * give user choice to choose from either automatic
 * or manual processing mode
 *
 * @author Cindy Yung
 * @version 1.0
 */


public class GolfInput extends JFrame
{
  ////////////////////// INSTANCE VARIABLES ////////////////////////////////

        private JButton buttonEnter;
        private JPanel ControlPanel;
        private JPanel BottomPanel;
        private JLabel valLabel, val2abel, val3label, Version;
        private JTextField val1, val2, val3;
        private GridLayout grid1;


 public GolfInput()
 {
       super("Golf Input");
       //setting the GUI with windows feel
//       WindowUtilities.setNativeLookAndFeel();
       grid1 = new GridLayout(2,1,5,0);
       Container container = getContentPane();
       container.setLayout(grid1);

       ControlPanel = new JPanel();
       Border ControlPanelBorder = BorderFactory.createTitledBorder("Golf Value ");
       ControlPanel.setBorder(ControlPanelBorder);
       valLabel = new JLabel("Please enter Speed (m/s)");
       ControlPanel.add(valLabel);
```

```
        ControlPanel.setLayout(new GridLayout(2,2,5,0));
        val1 = new JTextField(10);
        ControlPanel.add(val1);

        ControlPanel.setLayout(new GridLayout(3,3,5,0));
        val2abel = new JLabel("Pleae enter Vertical angle (deg)");
        ControlPanel.add(val2abel);

        ControlPanel.setLayout(new GridLayout(3,3,5,0));
        val2 = new JTextField(10);
        ControlPanel.add(val2);

       ControlPanel.setLayout(new GridLayout(4, 4, 5, 0));
        val3label = new JLabel("Pleae enter Right/left (deg)");
        ControlPanel.add(val3label);

        ControlPanel.setLayout(new GridLayout(4, 4, 5, 0));
        val3 = new JTextField(10);
        ControlPanel.add(val3);


        ControlPanel.setLayout(new GridLayout(5,5,5,0));
        buttonEnter = new JButton("Enter");
        buttonEnter.addActionListener(new ActionListener() {
          public void actionPerformed(ActionEvent ae) {
//            dispose();
            double text = Double.valueOf(val1.getText().trim()).doubleValue();
            double text2 = Double.valueOf(val2.getText().trim()).doubleValue();
            double text3 =  Double.valueOf(val3.getText().trim()).doubleValue();
            new ReadWriteFile(text, text2, text3);
        }
        });
        ControlPanel.add(buttonEnter);

        BottomPanel = new JPanel();
        BottomPanel.setLayout(new GridLayout(1,1));

        Version = new JLabel("Version 0.1 Created by Cindy Yung");
        BottomPanel.add(Version);

       // container.add(TopPanel);
        container.add(ControlPanel);
        container.add(BottomPanel);

        setSize(450,270);
        Dimension screen = Toolkit.getDefaultToolkit().getScreenSize();
        int x = (screen.width - 450) / 2;
        int y = (screen.height - 270) / 2;
        setLocation(x, y);
        setVisible(true);
    try {
      jbInit();
    }
    catch(Exception e) {
      e.printStackTrace();
    }
  }
 public static void main(String args[])
  {
        GolfInput application = new GolfInput();
        application.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  }
   private void jbInit() throws Exception {
   }
 }
```

Version 2 –ReadWriteFile.java Source code

```
import java.io.*;
import java.lang.Math.*;
import java.lang.String.*;

public class ReadWriteFile {

  public double preValA =0.00;
  public double preValB =0.00;
  public double preValC =0.00;

  public ReadWriteFile(){
  }

  public ReadWriteFile(double val, double val2, double val3){
    setContent(val, val2, val3);
  }

  static public void setContent(double speed, double vAngle, double rightLeftVal)
  {
    StringBuffer sb = new StringBuffer();
    double distance =0.00, height =0.00, rightLeft = 0.00;
    double preValA =0.00, preValB =0.00, preValC =0.00;



    System.out.println("New input ++++++++++++++++++ " + speed );
    System.out.println("New input 2 ++++++++++++++++ " + vAngle );
    System.out.println("New input 3 ++++++++++++++++ " + rightLeftVal );
    try {
       //FileWriter = new FileWriter
       BufferedWriter out = new BufferedWriter(new FileWriter("c:\\Fyp\\version2\\Golf.wrl",
false));
       sb.append("#VRML V2.0 utf8\n\n" +
            "NavigationInfo { headlight TRUE type [\"EXAMINE\", \"ANY\"]}  \n" +
            "Background { skyColor .333 .333 .333 }  \n"+
            "Viewpoint {       \n  " +
            "   position -10.0 8.0 25.0 \n   "+
            "   orientation -0.947 -10.6 -0.068 0.80 \n   "+
            "   description \"click!\"          \n"  +
            " }                        \n \n"+
            "   DEF ROOT Group {  \n"+
            "   children [   \n\n"+
            "   Transform {   \n"+
            "        children [                        \n"+
            "          DEF BOX Transform {             \n"+
            "            translation -4 0.5 0       \n"+
            "            children [           \n"+
            "                    Shape {          \n"+
            "                       appearance Appearance { material Material {} }    \n"+
            "                       geometry Sphere { radius 0.5}       \n"+
            "                    }                              \n"+
            "            ]                        \n"+
            "          }                        \n"+
            "       ]                       \n"+
            "     }                       \n"+
            "     Transform {          \n"+
            "       children [     \n"+
            "         Transform {                                          \n"+
            "           rotation 0 1 0 -.628                              \n"+
            "           children USE BOX                                 \n"+
            "         }                                          \n"+
            "       ]                                          \n"+
            "     }                                          \n"+
```

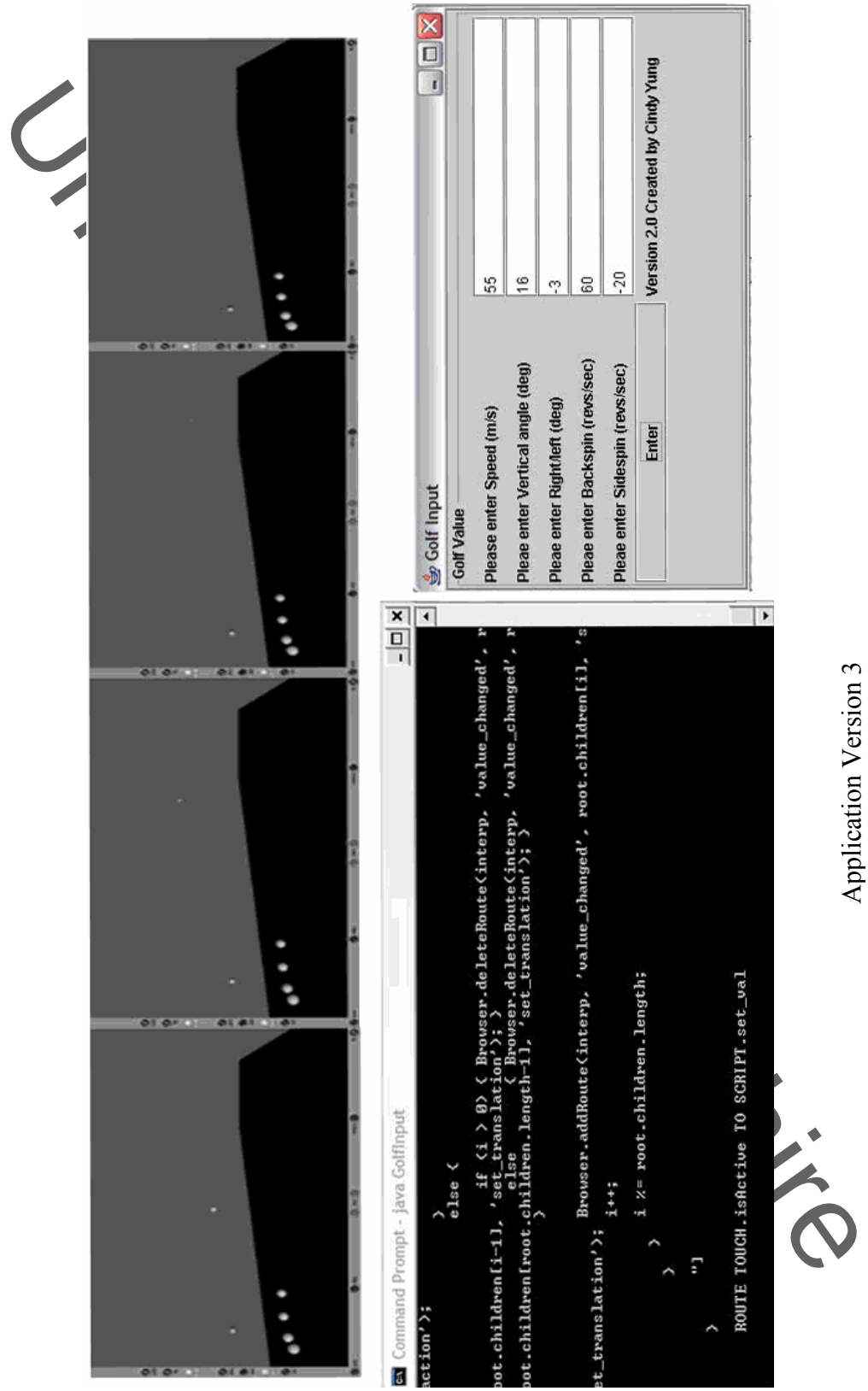
```
"                                                \n"+
"    Transform {                                   \n"+
"      children [                                \n"+
"        Transform {                             \n"+
"          rotation 0 1 0 -1.256  \n"+
"          children USE BOX                      \n"+
"        }                                  \n"+
"      ]                                  \n"+
"    }                                  \n"+
"                                  \n"+
"    Transform {                     \n"+
"      children [     \n"+
"        Transform {                                                          \n"+
"          rotation 0 1 0 -1.885                                              \n"+
"          children USE BOX                                                 \n"+
"        }                                                                \n"+
"      ]                                                               \n"+
"    }                                                              \n"+
"                                                                  \n"+
"    Transform {                                                     \n"+
"      children [                                                 \n"+
"        Transform {                                              \n"+
"          rotation 0 1 0 -2.513                                 \n"+
"          children USE BOX                                   \n"+
"        }                                        \n"+
"      ]                                      \n"+
"    }                                      \n"+
"                Shape {          \n"+
"                  appearance Appearance { material Material {diffuseColor 0.0 1.0
0.0} }     \n"+
"                  geometry Box { size 400 0.1 50}                        \n"+
"                }                                                  \n"+
"                                                                \n"+
"  ]                                                            \n"+
"}                                                             \n"+
"                                                            \n"+
"                                                              \n"+
"                                                            \n"+
"    Transform {                                               \n"+
"      translation -3 5 0                                    \n"+
"       children [                                     \n"+
"         DEF TOUCH TouchSensor {}      \n"+
"         Shape {                 \n"+
"           appearance Appearance { material Material {} }                \n"+
"           geometry Sphere { radius .25}                          \n"+
"         }                                                  \n"+
"       ]                                                  \n"+
"    }                                                   \n"+
"                                                      \n"+
"    DEF TIME TimeSensor {                                 \n"+
"      loop TRUE                                       \n"+
"    }                                          \n"+
"                                             \n"+
"    DEF PI PositionInterpolator {      \n"+
"       key [                          \n"+
" 0.00,                                \n"+
" 0.01,                                \n"+
" 0.02,                                \n"+
" 0.03,                                \n"+
" 0.04,                                \n"+
" 0.05,                                \n"+
" 0.06,                                \n"+
" 0.07,                                \n"+
" 0.08,                                \n"+
" 0.09,                                \n"+
" 0.1,                                \n"+
```

```
" 0.11,                    \n"+
" 0.12,                    \n"+
" 0.13,                    \n"+
" 0.14,                    \n"+
" 0.15,                    \n"+
" 0.16,                    \n"+
" 0.17,                    \n"+
" 0.18,                    \n"+
" 0.19,                    \n"+
" 0.2,                    \n"+
" 0.21,                    \n"+
" 0.22,                    \n"+
" 0.23,                    \n"+
" 0.24,                    \n"+
" 0.25,                    \n"+
" 0.26,                    \n"+
" 0.27,                    \n"+
" 0.28,                    \n"+
" 0.29,                    \n"+
" 0.3,                    \n"+
" 0.31,                    \n"+
" 0.32,                    \n"+
" 0.33,                    \n"+
" 0.34,                    \n"+
" 0.35,                    \n"+
" 0.36,                    \n"+
" 0.37,                    \n"+
" 0.38,                    \n"+
" 0.39,                    \n"+
" 0.4,                    \n"+
" 0.41,                    \n"+
" 0.42,                    \n"+
" 0.43,                    \n"+
" 0.44,                    \n"+
" 0.45,                    \n"+
" 0.46,                    \n"+
" 0.47,                    \n"+
" 0.48,                    \n"+
" 0.49,                    \n"+
" 0.5,                    \n"+
" 0.51,                    \n"+
" 0.52,                    \n"+
" 0.53,                    \n"+
" 0.54,                    \n"+
" 0.55,                    \n"+
" 0.56,                    \n"+
" 0.57,                    \n"+
" 0.58,                    \n"+
" 0.59,                    \n"+
" 0.6,                    \n"+
" 0.61,                    \n"+
" 0.62,                    \n"+
" 0.63,                    \n"+
" 0.64,                    \n"+
" 0.65,                    \n"+
" 0.66,                    \n"+
" 0.67,                    \n"+
" 0.68,                    \n"+
" 0.69,                    \n"+
" 0.7,                    \n"+
" 0.71,                    \n"+
" 0.72,                    \n"+
" 0.73,                    \n"+
" 0.74,                    \n"+
" 0.75,                    \n"+
```

```
 " 0.8,                                \n"+
" 0.81,                                \n"+
" 0.82,                                \n"+
" 0.83,                                \n"+
" 0.84,                                \n"+
" 0.85,                                \n"+
" 0.86,                                \n"+
" 0.87,                                \n"+
" 0.88,                                \n"+
" 0.89,                                \n"+
" 0.9,                                \n"+
" 0.91,                                \n"+
" 0.92,                                \n"+
" 0.93,                                \n"+
" 0.94,                                \n"+
" 0.95,                                \n"+
" 0.96,                                \n"+
" 0.97,                                \n"+
" 0.98,                                \n"+
" 0.99,                                \n"+
" 1                                \n"+
" ]                                \n"+
"  keyValue [ 0 0 0                        \n");

for(int i=0; i<100; i++)
{
  distance = calDistance(speed, vAngle, rightLeftVal);
  height = calHeight(speed, vAngle);
  rightLeft = calRightLeft(speed, vAngle, rightLeftVal);

  preValA = preValA+0.1*distance;
  preValB = preValB+height*0.1;
  preValC = preValC+rightLeft*0.1;

  if(i<99)
    sb.append(" "+preValA+" "+preValB+" "+preValC+", \n ");
  else
    sb.append(" "+preValA+" "+preValB+" "+preValC+"] ");

  System.out.println("Current Pos "+i+" distance = "+preValA+" height = "+preValB+
                  " Right/Left = "+ preValC);

}

sb.append("     }                                                \n"+
"                                        \n"+
"     DEF SCRIPT Script {                        \n"+
"       eventIn SFBool set_val                \n"+
"                                    \n"+
"       field SFBool   first  TRUE       \n"+
"       field SFInt32  i      0                    \n"+
"       field SFNode   timer  USE TIME             \n"+
"       field SFNode   interp USE PI            \n"+
"       field SFNode   root   USE ROOT        \n"+
"                                  \n"+
"       directOutput TRUE             \n"+
"       url [\"javascript:                          \n"+
"                                       \n"+
"         function set_val (clicked) {         \n"+
"           if (clicked) {                \n"+
"             if (first) {           \n"+
```

```
            "                first = false;   \n"+
            "                Browser.addRoute(timer, 'fraction_changed', interp, 'set_fraction');n"+
            "              }                                          \n"+
            "              else {                                                                  \n"+
            "                if (i > 0) { Browser.deleteRoute(interp, 'value_changed',
root.children[i-1], 'set_translation'); }  \n"+
            "                else       { Browser.deleteRoute(interp, 'value_changed',
root.children[root.children.length-1], 'set_translation'); }       \n"+
            "              }  \n"+
            "                                                                                  \n"+
            "              Browser.addRoute(interp, 'value_changed', root.children[i],
'set_translation');     \n"+
            "              i++;                                                \n"+
            "              i %= root.children.length;                              \n"+
            "            }                                                  \n"+
            "          }                                                  \n"+
            "                                                           \n"+
            "          \"]                                              \n"+
            "     }                                                \n"+
            "                                                   \n"+
            "     ROUTE TOUCH.isActive TO SCRIPT.set_val  \n");

          System.out.println(sb.toString());
          out.write(sb.toString());


      out.close();
    } catch (IOException e) {
      }
  }

  public static double calDistance(double x, double y, double z)
  {
    double dist = 0.00;

    dist = x*Math.cos(y/57.296)*Math.cos(z/57.296);
    return(dist);

  }

  public static double calHeight(double x, double y)
  {
    double height = 0.00;

    height = x*Math.sin(y/57.296);
    return(height);
  }

  public static double calRightLeft(double x, double y, double z)
  {
    double right = 0.00;

    right = x*Math.sin(z/57.296)*Math.cos(y/57.296);
    return(right);
  }
}
```

Version 3 - application



Application Version 3

Version 3 – GolfInpt.java Source code

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.*;

/**
 * <p>Final Year Project Code
 * <p>Main class to the image processing class
 * give user choice to choose from either automatic
 * or manual processing mode
 *
 * @author Cindy Yung
 * @version 2.0
 */

public class GolfInput extends JFrame
{
  ////////////////////// INSTANCE VARIABLES ////////////////////////////////

        private JButton buttonEnter;
        private JPanel ControlPanel;
        private JPanel BottomPanel;
        private JLabel valLabel, val2abel, val3label, m_backSpinLabel, m_sideSpinLabel, Version;
        private JTextField val1, val2, val3, m_backSpin, m_sideSpin;
        private GridLayout grid1;

 public GolfInput()
 {
       super("Golf Input");
       //setting the GUI with windows feel
           //WindowUtilities.setNativeLookAndFeel();
       //grid1 = new GridLayout(2,1,5,0);
       grid1 = new GridLayout(1,0,0,0);
       Container container = getContentPane();
       container.setLayout(grid1);

       ControlPanel = new JPanel();
       Border ControlPanelBorder = BorderFactory.createTitledBorder("Golf Value ");
       ControlPanel.setBorder(ControlPanelBorder);
       valLabel = new JLabel("Please enter Speed (m/s)");
       ControlPanel.add(valLabel);

       ControlPanel.setLayout(new GridLayout(2,2,5,0));
       val1 = new JTextField(10);
       ControlPanel.add(val1);

       ControlPanel.setLayout(new GridLayout(3,3,5,0));
       val2abel = new JLabel("Pleae enter Vertical angle (deg)");
       ControlPanel.add(val2abel);

       ControlPanel.setLayout(new GridLayout(3,3,5,0));
       val2 = new JTextField(10);
       ControlPanel.add(val2);

       ControlPanel.setLayout(new GridLayout(4, 4, 5, 0));
       val3label = new JLabel("Pleae enter Right/left (deg)");
       ControlPanel.add(val3label);

       ControlPanel.setLayout(new GridLayout(4, 4, 5, 0));
       val3 = new JTextField(10);
       ControlPanel.add(val3);
```

```
        ControlPanel.setLayout(new GridLayout(5, 5, 5, 0));
        m_backSpinLabel = new JLabel("Pleae enter Backspin (revs/sec)");
        ControlPanel.add(m_backSpinLabel);

        ControlPanel.setLayout(new GridLayout(5, 5, 5, 0));
        m_backSpin = new JTextField(10);
        ControlPanel.add(m_backSpin);

        ControlPanel.setLayout(new GridLayout(6, 6, 5, 0));
        m_sideSpinLabel = new JLabel("Pleae enter Sidespin (revs/sec)");
        ControlPanel.add(m_sideSpinLabel);

        ControlPanel.setLayout(new GridLayout(6, 6, 5, 0));
        m_sideSpin = new JTextField(10);
        ControlPanel.add(m_sideSpin);

        ControlPanel.setLayout(new GridLayout(7,7,5,0));
        buttonEnter = new JButton("Enter");
        buttonEnter.addActionListener(new ActionListener() {

          public void actionPerformed(ActionEvent ae) {
//            dispose();
           double text = Double.valueOf(val1.getText().trim()).doubleValue();
           double text2 = Double.valueOf(val2.getText().trim()).doubleValue();
           double text3 =  Double.valueOf(val3.getText().trim()).doubleValue();
           double text4 = Double.valueOf(m_backSpin.getText().trim()).doubleValue();
           double text5 =  Double.valueOf(m_sideSpin.getText().trim()).doubleValue();
           new ReadWriteFile(text, text2, text3, text4, text5);
        }
        });
        ControlPanel.add(buttonEnter);

       // BottomPanel = new JPanel();
       // BottomPanel.setLayout(new GridLayout(1,1));
            ControlPanel.setLayout(new GridLayout(8,8,5,0));
        Version = new JLabel("Version 2.0 Created by Cindy Yung");
        ControlPanel.add(Version);


       // container.add(TopPanel);
        container.add(ControlPanel);
        //container.add(BottomPanel);

        setSize(450,270);
        Dimension screen = Toolkit.getDefaultToolkit().getScreenSize();
        int x = (screen.width - 450) / 2;
        int y = (screen.height - 270) / 2;
        setLocation(x, y);
        setVisible(true);
        try {
          jbInit();
        }
        catch(Exception e) {
          e.printStackTrace();
        }
  }
   private void jbInit() throws Exception {
   }
  public static void main(String args[])
  {
        GolfInput application = new GolfInput();
        application.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  }
 }
```

Version 3 – ReadWriteFile.java Source code

```
import java.io.*;
import java.lang.Math.*;
import java.lang.String.*;
import java.text.NumberFormat.*;
import java.text.DecimalFormat.*;

public class ReadWriteFile {

  public double preValA =0.00;
  public double preValB =0.00;
  public double preValC =0.00;

  public ReadWriteFile(){
  }

  public ReadWriteFile(double val, double val2, double val3, double val4, double val5){
    setContent(val, val2, val3, val4, val5);
  }

  static public void setContent(double speed, double vAngle, double rightLeftVal, double
backSpin, double sideSpin)
  {
    StringBuffer sb = new StringBuffer();
    double distance =0.00, height =0.00, rightLeft = 0.00;
    double vA =0.00, vB =0.00, vC =0.00, v = 0.00, alpha = 0.00;
    double preDistance =0.00, preHeight = 0.00, preRightLeft = 0.00;



    System.out.println("New input ++++++++++++++++++ " + speed );
    System.out.println("New input 2 +++++++++++++++++ " + vAngle );
    System.out.println("New input 3 +++++++++++++++++ " + rightLeftVal );
    try {
       //FileWriter = new FileWriter
       BufferedWriter out = new BufferedWriter(new FileWriter("c:\\Fyp\\version3\\Golf.wrl",
false));
       sb.append("#VRML V2.0 utf8\n\n" +
            "NavigationInfo { headlight TRUE type [\"EXAMINE\", \"ANY\"]}  \n" +
            "Background { skyColor .333 .333 .333 }  \n"+
            "Viewpoint {       \n  " +
            "   position -10.0 8.0 25.0\n   "+
            "   orientation -0.947 -10.6 -0.068 0.80  \n   "+
            "   description \"click!\"          \n"  +
            " }                        \n \n"+
            "   DEF ROOT Group {  \n"+
            "   children [   \n\n"+
            "   Transform {   \n"+
            "        children [                        \n"+
            "          DEF BOX Transform {             \n"+
            "            translation -4 0.5 0       \n"+
            "            children [           \n"+
            "                    Shape {          \n"+
            "                       appearance Appearance { material Material {} }    \n"+
            "                       geometry Sphere { radius 0.5}       \n"+
            "                    }                                \n"+
            "            ]                        \n"+
            "          }                        \n"+
            "       ]                        \n"+
            "     }                        \n"+
            "     Transform {          \n"+
            "       children [     \n"+
            "         Transform {                                        \n"+
            "           rotation 0 1 0 -.628                            \n"+
```

```
"            children USE BOX                                    \n"+
"          }                                                     \n"+
"        ]                                                       \n"+
"      }                                                         \n"+
"                                                                \n"+
"      Transform {                                               \n"+
"        children [                                              \n"+
"          Transform {                                           \n"+
"            rotation 0 1 0 -1.256  \n"+
"            children USE BOX                                    \n"+
"          }                                                     \n"+
"        ]                                                       \n"+
"      }                                                         \n"+
"                                                                \n"+
"      Transform {                                               \n"+
"        children [     \n"+
"          Transform {                                                       \n"+
"            rotation 0 1 0 -1.885                                           \n"+
"            children USE BOX                                                \n"+
"          }                                                                 \n"+
"        ]                                                                   \n"+
"      }                                                                     \n"+
"                                                                            \n"+
"      Transform {                                                           \n"+
"        children [                                                          \n"+
"          Transform {                                                       \n"+
"            rotation 0 1 0 -2.513                                           \n"+
"            children USE BOX                                                \n"+
"          }                                                                 \n"+
"        ]                                                                   \n"+
"      }                                                                     \n"+
"                    Shape {          \n"+
"                      appearance Appearance { material Material {diffuseColor 0.0 1.0
0.0} }     \n"+
"                      geometry Box { size 400 0.1 50}                       \n"+
"                    }                                                       \n"+
"                                                                            \n"+
"  ]                                                                         \n"+
"}                                                                           \n"+
"                                                                            \n"+
"                                                                            \n"+
"                                                                            \n"+
"      Transform {                                                           \n"+
"        translation -3 5 0                                                  \n"+
"         children [                                                         \n"+
"           DEF TOUCH TouchSensor {}      \n"+
"           Shape {                 \n"+
"             appearance Appearance { material Material {} }                \n"+
"             geometry Sphere { radius .25}                                 \n"+
"           }                                                                \n"+
"         ]                                                                  \n"+
"      }                                                                     \n"+
"                                                                            \n"+
"      DEF TIME TimeSensor {                                                 \n"+
"                            cycleInterval 5.0
          \n"+
"        loop TRUE                                                           \n"+
"      }                                                                     \n"+
"                                                                            \n"+
"      DEF PI PositionInterpolator {      \n"+
"         key [                                                              \n"+
" 0.00,                                                                      \n"+
" 0.01,                                                                      \n"+
" 0.02,                                                                      \n"+
```

```
" 0.03,                    \n"+
" 0.04,                    \n"+
" 0.05,                    \n"+
" 0.06,                    \n"+
" 0.07,                    \n"+
" 0.08,                    \n"+
" 0.09,                    \n"+
" 0.1,                    \n"+
" 0.11,                    \n"+
" 0.12,                    \n"+
" 0.13,                    \n"+
" 0.14,                    \n"+
" 0.15,                    \n"+
" 0.16,                    \n"+
" 0.17,                    \n"+
" 0.18,                    \n"+
" 0.19,                    \n"+
" 0.2,                    \n"+
" 0.21,                    \n"+
" 0.22,                    \n"+
" 0.23,                    \n"+
" 0.24,                    \n"+
" 0.25,                    \n"+
" 0.26,                    \n"+
" 0.27,                    \n"+
" 0.28,                    \n"+
" 0.29,                    \n"+
" 0.3,                    \n"+
" 0.31,                    \n"+
" 0.32,                    \n"+
" 0.33,                    \n"+
" 0.34,                    \n"+
" 0.35,                    \n"+
" 0.36,                    \n"+
" 0.37,                    \n"+
" 0.38,                    \n"+
" 0.39,                    \n"+
" 0.4,                    \n"+
" 0.41,                    \n"+
" 0.42,                    \n"+
" 0.43,                    \n"+
" 0.44,                    \n"+
" 0.45,                    \n"+
" 0.46,                    \n"+
" 0.47,                    \n"+
" 0.48,                    \n"+
" 0.49,                    \n"+
" 0.5,                    \n"+
" 0.51,                    \n"+
" 0.52,                    \n"+
" 0.53,                    \n"+
" 0.54,                    \n"+
" 0.55,                    \n"+
" 0.56,                    \n"+
" 0.57,                    \n"+
" 0.58,                    \n"+
" 0.59,                    \n"+
" 0.6,                    \n"+
" 0.61,                    \n"+
" 0.62,                    \n"+
" 0.63,                    \n"+
" 0.64,                    \n"+
" 0.65,                    \n"+
" 0.66,                    \n"+
" 0.67,                    \n"+
```

```
      " 0.67,                          \n"+
      " 0.68,                          \n"+
      " 0.69,                          \n"+
      " 0.7,                          \n"+
      " 0.71,                          \n"+
      " 0.72,                          \n"+
      " 0.73,                          \n"+
      " 0.74,                          \n"+
      " 0.75,                          \n"+
      " 0.76,                          \n"+
      " 0.77,                          \n"+
      " 0.78,                          \n"+
      " 0.79,                          \n"+
      " 0.8,                          \n"+
      " 0.81,                          \n"+
      " 0.82,                          \n"+
      " 0.83,                          \n"+
      " 0.84,                          \n"+
      " 0.85,                          \n"+
      " 0.86,                          \n"+
      " 0.87,                          \n"+
      " 0.88,                          \n"+
      " 0.89,                          \n"+
      " 0.9,                          \n"+
      " 0.91,                          \n"+
      " 0.92,                          \n"+
      " 0.93,                          \n"+
      " 0.94,                          \n"+
      " 0.95,                          \n"+
      " 0.96,                          \n"+
      " 0.97,                          \n"+
      " 0.98,                          \n"+
      " 0.99,                          \n"+
      " 1                          \n"+
      " ]                          \n"+
      "  keyValue [ 0 0 0,                    \n");
//initial calculation
vA = calInitialVa(speed, vAngle, rightLeftVal);
vB = calInitialVb(speed, vAngle);
vC = calInitialVc(speed, vAngle, rightLeftVal);
alpha = calAlpha(sideSpin, backSpin);

for(int i=0; i<100; i++)
{

     if(i>0)
     {
   if (height <0)
   {
        vA = 0.0;
    vB = 0.0;
    vC = 0.0;
   }
    else
    {
    vA = calVa(vA, vB, vC, v, alpha);
    vB = calVb(vA, vB, v);
    vC = calVc(vA, vC, v, alpha);
    }
  }


  v = calV(vA, vB, vC );

  distance = distance + 0.1 * vA;
```

```
        height = height + vB * 0.1;
        rightLeft = rightLeft + vC * 0.1;

        if (i < 99)
          sb.append(" " + distance + " " + height + " " + rightLeft + ", \n");
        else
          sb.append(" " + distance + " " + height + " " + rightLeft + "] \n");

        System.out.println("Current Pos " + i + " distance = " + distance +
                       " height = " + height +
                       " Right/Left = " + rightLeft);

        // preDistance = distance;
        //preHeight = height;
        //preRightLeft = rightLeft;
      }

      sb.append("     }                                                    \n"+
      "                                                  \n"+
      "     DEF SCRIPT Script {                          \n"+
      "       eventIn SFBool set_val                     \n"+
      "                                        \n"+
      "       field SFBool   first  TRUE       \n"+
      "       field SFInt32  i      0                    \n"+
      "       field SFNode   timer  USE TIME             \n"+
      "       field SFNode   interp USE PI               \n"+
      "       field SFNode   root   USE ROOT             \n"+
      "                                      \n"+
      "       directOutput TRUE                \n"+
      "       url [\"javascript:                         \n"+
      "                                            \n"+
      "         function set_val (clicked) {             \n"+
      "           if (clicked) {                   \n"+
      "             if (first) {                \n"+
      "               first = false;   \n"+
      "               Browser.addRoute(timer, 'fraction_changed', interp, 'set_fraction'); \n"+
      "             }                                    \n"+
      "             else {                                                          \n"+
      "               if (i > 0) { Browser.deleteRoute(interp, 'value_changed', root.children[i-1],
'set_translation'); }  \n"+
      "               else       { Browser.deleteRoute(interp, 'value_changed',
root.children[root.children.length-1], 'set_translation'); }       \n"+
      "             }  \n"+
      "                                                                          \n"+
      "             Browser.addRoute(interp, 'value_changed', root.children[i],
'set_translation');     \n"+
      "             i++;                                       \n"+
      "             i %= root.children.length;                     \n"+
      "           }                                          \n"+
      "         }                                          \n"+
      "                                                    \n"+
      "         \"]                                        \n"+
      "     }                                        \n"+
      "                                          \n"+
      "     ROUTE TOUCH.isActive TO SCRIPT.set_val  \n");

     System.out.println(sb.toString());
     out.write(sb.toString());


      out.close();
   } catch (IOException e) {
    }
  }
```

```
  private static double calInitialVa(double x, double y, double z)
  {
    double dist = 0.00;

    dist = x*Math.cos(y/57.296)*Math.cos(z/57.296);
    return(dist);

  }

  private static double calInitialVb(double x, double y)
  {
    return(x*Math.sin(y/57.296));
  }

  private static double calInitialVc(double x, double y, double z)
  {
    return(x*Math.sin(z/57.296)*Math.cos(y/57.296));
  }

  private static double calV(double VA, double VB, double VC)
  {
    return(Math.sqrt(VA*VA+VB*VB+VC*VC));
  }

  private static double calAlpha(double SideSpin, double BackSpin)
  {
    return(Math.atan(SideSpin/BackSpin));
  }

  private static double calVa(double VA, double VB, double VC, double V, double Alpha)
  {
    return((VA-0.0019*V*(0.25*VA-0.2*(VC*Alpha-VB))));
  }

  private static double calVb(double VA, double VB, double V)
  {
    return((VB-0.98-0.0019*V*(0.25*VB-0.2*VA)));
  }

  private static double calVc(double VA,  double VC, double V, double Alpha)
  {
    return((VC-0.0019*V*(0.25*VC+0.2*VA*Alpha)));
  }


}
```

shire

Appendix C

Final application screen shot

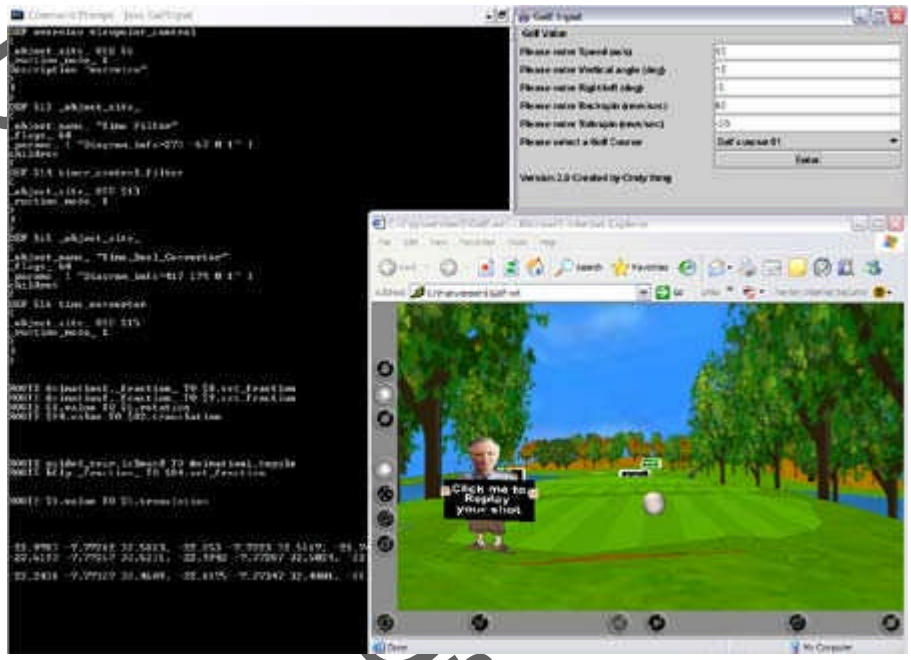


Figure 44:Final application

Source Code

In this section, the source code file will be shown as follows:

- GolfInput.java (Interface)
- ReadWriteFile.java (the main program)
- GolfInput.class
- GolfInput$1.class
- GolfInput$2.class
- ReadWriteFile.class

For the VRML golf courses source codes, it can be found in the CD enclosed.

University of Hertfordshire

GolfInput.java - Interface

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.*;
/**
 * <p>Final Year Project Code
 * <p>Main class to the image processing class
 * give user choice to choose from either automatic
 * or manual processing mode
 *
 * @author Cindy Yung
 * @version 2.0
 */
public class GolfInput extends JFrame
{
  ////////////////////// INSTANCE VARIABLES ////////////////////////////////
        private JButton buttonEnter;
        private JPanel ControlPanel;
        private JPanel BottomPanel;
        private JLabel valLabel, val2abel, val3label, m_backSpinLabel,
                m_sideSpinLabel, Version, golfSelection;
        private JTextField val1, val2, val3, m_backSpin, m_sideSpin;
        private JComboBox golfChoiceBox;
        private GridLayout grid1;
        private String golfCourseNames[] = {"Golf course 01", "Golf course 02", "Golf course
03"};
        private Icon icons[] = {new ImageIcon(golfCourseNames[0]),
                        new ImageIcon(golfCourseNames[1]), new
ImageIcon(golfCourseNames[2])};

        private int golfCourseChoice = 0;

 public GolfInput()
 {
       super("Golf Input");
       //setting the GUI with windows feel

       grid1 = new GridLayout(1,0,0,0);
       Container container = getContentPane();
       container.setLayout(grid1);

       ControlPanel = new JPanel();
       Border ControlPanelBorder = BorderFactory.createTitledBorder("Golf Value ");
       ControlPanel.setBorder(ControlPanelBorder);
       valLabel = new JLabel("Please enter Speed (m/s)");
       ControlPanel.add(valLabel);

       ControlPanel.setLayout(new GridLayout(2,2,5,0));
       val1 = new JTextField(10);
       ControlPanel.add(val1);
```

University Of Hertfordshire

```
ControlPanel.setLayout(new GridLayout(3,3,5,0));
val2abel = new JLabel("Please enter Vertical angle (deg)");
ControlPanel.add(val2abel);

ControlPanel.setLayout(new GridLayout(3,3,5,0));
val2 = new JTextField(10);
ControlPanel.add(val2);

ControlPanel.setLayout(new GridLayout(4, 4, 5, 0));
val3label = new JLabel("Please enter Right/left (deg)");
ControlPanel.add(val3label);

ControlPanel.setLayout(new GridLayout(4, 4, 5, 0));
val3 = new JTextField(10);
ControlPanel.add(val3);

ControlPanel.setLayout(new GridLayout(5, 5, 5, 0));
m_backSpinLabel = new JLabel("Please enter Backspin (revs/sec)");
ControlPanel.add(m_backSpinLabel);

ControlPanel.setLayout(new GridLayout(5, 5, 5, 0));
m_backSpin = new JTextField(10);
ControlPanel.add(m_backSpin);

ControlPanel.setLayout(new GridLayout(6, 6, 5, 0));
m_sideSpinLabel = new JLabel("Please enter Sidespin (revs/sec)");
ControlPanel.add(m_sideSpinLabel);

ControlPanel.setLayout(new GridLayout(6, 6, 5, 0));
m_sideSpin = new JTextField(10);
ControlPanel.add(m_sideSpin);

ControlPanel.setLayout(new GridLayout(7,7,5,0));
golfSelection = new JLabel("Please select a Golf Course");
ControlPanel.add(golfSelection);

// set up JComboBox and register its event handler
ControlPanel.setLayout(new GridLayout(7,7,5,0));
golfChoiceBox = new JComboBox(golfCourseNames);
golfChoiceBox.setMaximumRowCount(3);
golfChoiceBox.addItemListener(

new ItemListener() {
   // handle JComboBox event
   public void itemStateChanged(ItemEvent event) {
      // determine whether check box selected
      if (event.getStateChange() == ItemEvent.SELECTED)
         golfSelection.setIcon(icons[golfChoiceBox.getSelectedIndex()]);
         golfCourseChoice = golfChoiceBox.getSelectedIndex();
   }
}
); // end call to addItemListener

ControlPanel.add(golfChoiceBox);
```

University Of Hertfordshire

```
        ControlPanel.setLayout(new GridLayout(7,7,5,0));
        // set up Jlabel tp display ImageIcons
        golfSelection = new JLabel(icons[0]);
        ControlPanel.add(golfSelection);

        setSize(350, 100);
        setVisible(true);

        ControlPanel.setLayout(new GridLayout(8,8,5,0));
        buttonEnter = new JButton("Enter");
        buttonEnter.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent ae) {

            double text = Double.valueOf(val1.getText().trim()).doubleValue();
            double text2 = Double.valueOf(val2.getText().trim()).doubleValue();
            double text3 =  Double.valueOf(val3.getText().trim()).doubleValue();
            double text4 = Double.valueOf(m_backSpin.getText().trim()).doubleValue();
            double text5 =  Double.valueOf(m_sideSpin.getText().trim()).doubleValue();
            System.out.println("Golf Course Choice is : " + golfCourseChoice);
            new ReadWriteFile(text, text2, text3, text4, text5, golfCourseChoice);
        }
    }
);
        ControlPanel.add(buttonEnter);
        ControlPanel.setLayout(new GridLayout(9,9,5,0));
        Version = new JLabel("Version 2.0 Created by Cindy Yung");
        ControlPanel.add(Version);
        container.add(ControlPanel);
        setSize(450,270);
        Dimension screen = Toolkit.getDefaultToolkit().getScreenSize();
        int x = (screen.width - 450) / 2;
        int y = (screen.height - 270) / 2;
        setLocation(x, y);
        setVisible(true);
        try {
          jbInit();
        }
        catch(Exception e) {
          e.printStackTrace();
        }
 }
  private void jbInit() throws Exception {
  }
 public static void main(String args[])
 {
        GolfInput application = new GolfInput();
        application.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
 }

}
```

University Of Hertfordshire

ReadWriteFile.java – The main program

```
import java.io.*;
import java.lang.Math.*;
import java.lang.String.*;
import java.text.NumberFormat.*;
import java.text.DecimalFormat.*;

public class ReadWriteFile {

  private double preValA =0.00;
  private double preValB =0.00;
  private double preValC =0.00;
  private int golfCourseChoice = 0;

  public ReadWriteFile(){
  }
// Iink to GolfInput.java
  public ReadWriteFile(double val, double val2, double val3, double val4, double val5, int
val6)
  {
   golfCourseChoice = val6;
   setContent(val, val2, val3, val4, val5);
  }

  private void setContent(double speed, double vAngle, double rightLeftVal, double
backSpin, double sideSpin)
  {
   StringBuffer sb = new StringBuffer();
   StringBuffer content = new StringBuffer();
   String orignalFileContent = "";
   String newContent = "";

   double distance =0.00, height =0.00, rightLeft = 0.00;
   double vA =0.00, vB =0.00, vC =0.00, v = 0.00, alpha = 0.00;
   double preDistance =0.00, preHeight = 0.00, preRightLeft = 0.00;

   System.out.println("New input ++++++++++++++++++ " + speed );
   System.out.println("New input 2 ++++++++++++++++ " + vAngle );
   System.out.println("New input 3 ++++++++++++++++ " + rightLeftVal );
   // FileWriter = new FileWriter
   // Write Golf.wrl with the new values in the specific location
   try {

        BufferedWriter out = new BufferedWriter(new FileWriter(
            "c:\\FYP\\version5\\Golf.wrl", false));

      //initial calculation
      vA = calInitialVa(speed, vAngle, rightLeftVal);
      vB = calInitialVb(speed, vAngle);
      vC = calInitialVc(speed, vAngle, rightLeftVal);
      alpha = calAlpha(sideSpin, backSpin);
```

University Of Hertfordshire

```
      for(int i=0; i<100; i++)
      {
       // If height < 0 VA,VB,VC will = to 0 else calculate this
           if(i>0)
           {
         if (height <0)
         {
          vA = 0.0;
          vB = 0.0;
          vC = 0.0;
         }
          else
          {
          vA = calVa(vA, vB, vC, v, alpha);
          vB = calVb(vA, vB, v);
          vC = calVc(vA, vC, v, alpha);
          }
        }


        v = calV(vA, vB, vC );

        distance = distance + 0.1 * vA;
        height = height + vB * 0.1;
        rightLeft = rightLeft + vC * 0.1;

        if (i < 99)
          sb.append(" " + rightLeft + " " + height + " " + distance + ", \n");
        else
          sb.append(" " + rightLeft + " " + height + " " + distance + " \n");

        System.out.println("Current Pos " + i + " distance = " + distance +
                           " height = " + height +
                           " Right/Left = " + rightLeft);
      }

// Read the original VRML file, until  it reaches the line "REPLACECODEHERE".
// This is where the calculated data from the trajectory formulas will be placed.
       orignalFileContent = getContents();
       newContent = replace(orignalFileContent, "REPLACECODEHERE", sb.toString());

      System.out.println(newContent);
      out.write(newContent);
       out.close();
    } catch (IOException e) {
       System.out.println("Error in saving file " + e);
     }
  }

  private String getContents() {
   //...checks on Golftest are elided
   StringBuffer contents = new StringBuffer();

   //declared here only to make visible to finally clause
   BufferedReader input = null;
```

University Of Hertfordshire

```
  try {
    //use buffering
    //this implementation reads one line at a time
    //FileReader always assumes default encoding is OK!
    if(golfCourseChoice == 0)
       input = new BufferedReader( new FileReader("C:\\FYP\\version5\\golf1_V13.wrl"));
    else if(golfCourseChoice == 1)
       input = new BufferedReader( new FileReader("C:\\FYP\\version5\\golf2_V09.wrl"));
    else if(golfCourseChoice == 2)
       input = new BufferedReader( new FileReader("C:\\FYP\\version5\\golf3_V04.wrl"));

    String line = null; //not declared within while loop
    while (( line = input.readLine()) != null){
      contents.append(line);
      contents.append(System.getProperty("line.separator"));
    }
  }
  catch (FileNotFoundException ex) {
    ex.printStackTrace();
  }
  catch (IOException ex){
    ex.printStackTrace();
  }
  finally {
    try {
      if (input!= null) {
        //flush and close both "input" and its underlying FileReader
        input.close();
      }
    }
    catch (IOException ex) {
      ex.printStackTrace();
    }
  }
  return contents.toString();
}

  private static String replace(final String aInput, final String aOldPattern, final String
aNewPattern)
  {
    return aInput.replaceAll(aOldPattern, aNewPattern);
  }
// calInitial
  private static double calInitialVa(double x, double y, double z)
  {
    double dist = 0.00;

    dist = x*Math.cos(y/57.296)*Math.cos(z/57.296)*-1;
    return(dist);

  }

  private static double calInitialVb(double x, double y)
  {
    return(x*Math.sin(y/57.296));
```

University Of Hertfordshire

```
  }

  private static double calInitialVc(double x, double y, double z)
  {
   return(x*Math.sin(z/57.296)*Math.cos(y/57.296));
  }

  private static double calV(double VA, double VB, double VC)
  {
   return(Math.sqrt(VA*VA+VB*VB+VC*VC));
  }

  private static double calAlpha(double SideSpin, double BackSpin)
  {
   return(Math.atan(SideSpin/BackSpin));
  }

  private static double calVa(double VA, double VB, double VC, double V, double Alpha)
  {
   return((VA-0.0019*V*(0.25*VA-0.2*(VC*Alpha-VB))));
  }

  private static double calVb(double VA, double VB, double V)
  {
   return((VB-0.98-0.0019*V*(0.25*VB-0.2*VA)));
  }

  private static double calVc(double VA,  double VC, double V, double Alpha)
  {
   return((VC-0.0019*V*(0.25*VC+0.2*VA*Alpha)));
  }
}
```

University Of Hertfordshire

Implemented structure of GolfInput.class

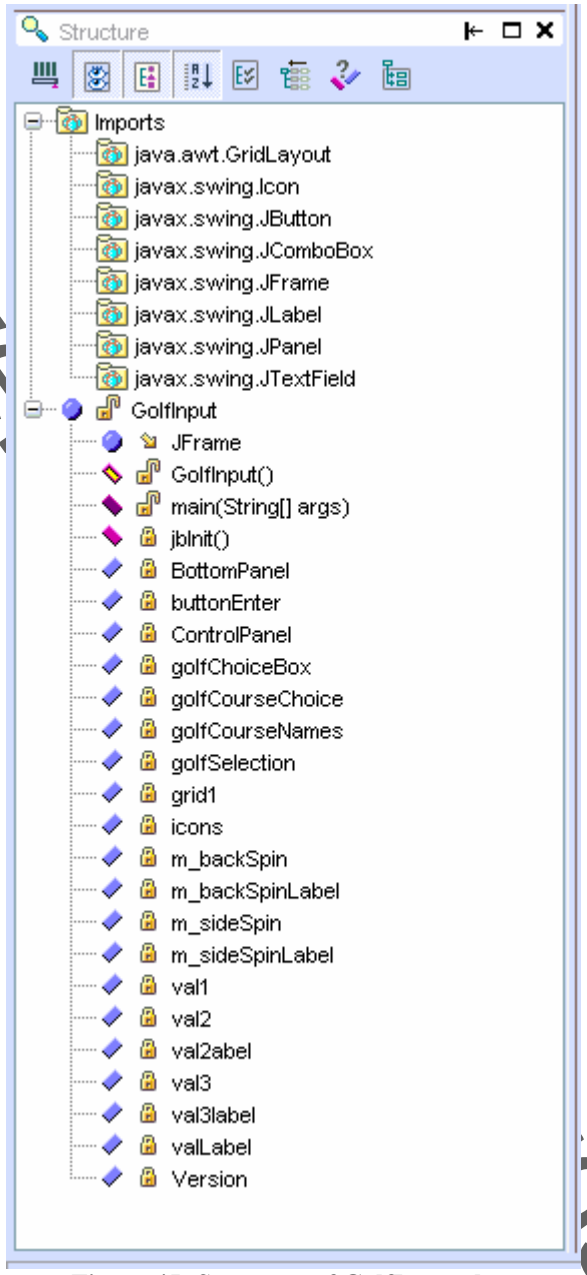


Figure 45: Structure of GolfInput.class

University of Hertfordshire

Source code - GolfInput.class

```
// JBuilder API Decompiler stub source generated from class file
// 21-Apr-2006
// -- implementation of methods is not available

// Imports
import java.awt.GridLayout;
import javax.swing.Icon;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class GolfInput extends JFrame {

  // Fields
  private JButton buttonEnter;
  private JPanel ControlPanel;
  private JPanel BottomPanel;
  private JLabel valLabel;
  private JLabel val2abel;
  private JLabel val3label;
  private JLabel m_backSpinLabel;
  private JLabel m_sideSpinLabel;
  private JLabel Version;
  private JLabel golfSelection;
  private JTextField val1;
  private JTextField val2;
  private JTextField val3;
  private JTextField m_backSpin;
  private JTextField m_sideSpin;
  private JComboBox golfChoiceBox;
  private GridLayout grid1;
  private String[] golfCourseNames;
  private Icon[] icons;
  private int golfCourseChoice;

  // Constructors
  public GolfInput() { }

  // Methods
  private void jbInit() throws Exception { }
  public static void main(String[] args) { }
}
```

University Of Hertfordshire

Source code - GolfInput$1.class

```
// JBuilder API Decompiler stub source generated from class file
// 21-Apr-2006
// -- implementation of methods is not available

// Imports
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;

class 1 implements ItemListener {

  // Constructors
  1(GolfInput null) { }

  // Methods
  public void itemStateChanged(ItemEvent event) { }
}
```

Source code - GolfInput$2.class

```
// JBuilder API Decompiler stub source generated from class file
// 21-Apr-2006
// -- implementation of methods is not available

// Imports
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

class 2 implements ActionListener {

  // Constructors
  2(GolfInput null) { }

  // Methods
  public void actionPerformed(ActionEvent ae) { }
}
```

University Of Hertfordshire

Implemented structure structure of ReadWriteFile.class

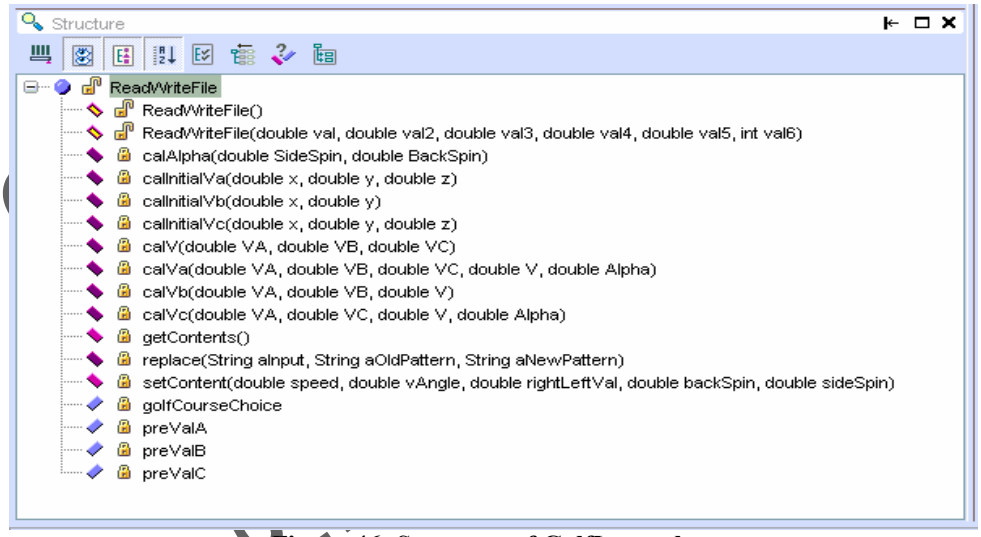

Structure
ReadWriteFile
ReadWriteFile()
ReadWriteFile(double val, double val2, double val3, double val4, double val5, int val6)
calAlpha(double SideSpin, double BackSpin)
callnitialVa(double x, double y, double z)
callnitialVb(double x, double y)
callnitialVc(double x, double y, double z)
calV(double VA, double VB, double VC)
calVa(double VA, double VB, double VC, double V, double Alpha)
calVb(double VA, double VB, double V)
calVc(double VA, double VC, double V, double Alpha)
getContents()
replace(String aInput, String aOldPattern, String aNewPattern)
setContent(double speed, double vAngle, double rightLeftVal, double backSpin, double sideSpin)
golfCourseChoice
preValA
preValB
preValC

Figure 46: Structure of GolfInput.class

Source code - ReadWriteFile.class

```
// JBuilder API Decompiler stub source generated from class file
// 21-Apr-2006
// -- implementation of methods is not available
public class ReadWriteFile {
  // Fields
  private double preValA;
  private double preValB;
  private double preValC;
  private int golfCourseChoice;
  // Constructors
  public ReadWriteFile() { }
  public ReadWriteFile(double val, double val2, double val3, double val4, double val5, int
val6) { }
  // Methods
  private void setContent(double speed, double vAngle, double rightLeftVal, double
backSpin, double sideSpin) { }
  private String getContents() { return null;}
  private static String replace(String aInput, String aOldPattern, String aNewPattern) { return
null;}
  private static double calInitialVa(double x, double y, double z) { return 0.0;}
  private static double calInitialVb(double x, double y) { return 0.0;}
  private static double calInitialVc(double x, double y, double z) { return 0.0;}
  private static double calV(double VA, double VB, double VC) { return 0.0;}
  private static double calAlpha(double SideSpin, double BackSpin) { return 0.0;}
  private static double calVa(double VA, double VB, double VC, double V, double Alpha)
{ return 0.0;}
  private static double calVb(double VA, double VB, double V) { return 0.0;}
  private static double calVc(double VA, double VC, double V, double Alpha) { return
0.0;}
}
```

University Of Hertfordshire

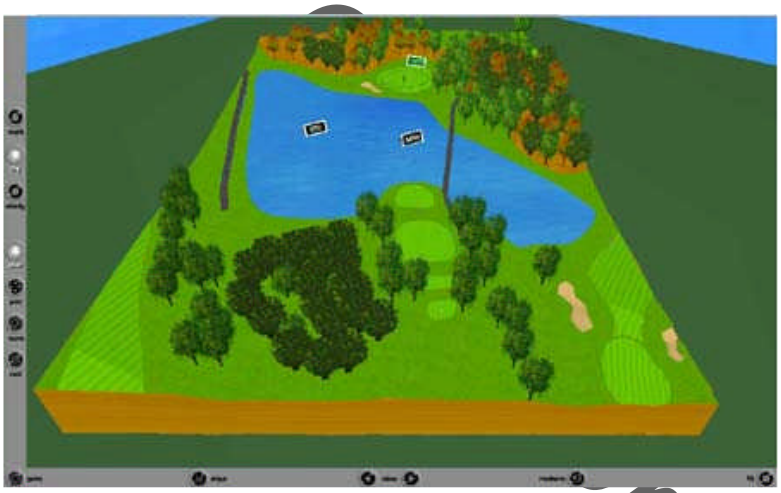
Appendix D

Screen Shots of all VRML Golf courses

Golf Course 01

Tee low view point

Figure 47: Golf course 01 – Tee low view



Tee high view point

Figure 48: Golf course 01 – Tee high view

Bird eye view

Note: the golf course view is rotated 90 degrees from the original viewpoint above

Figure 49: Golf course 01 – Bird eye view

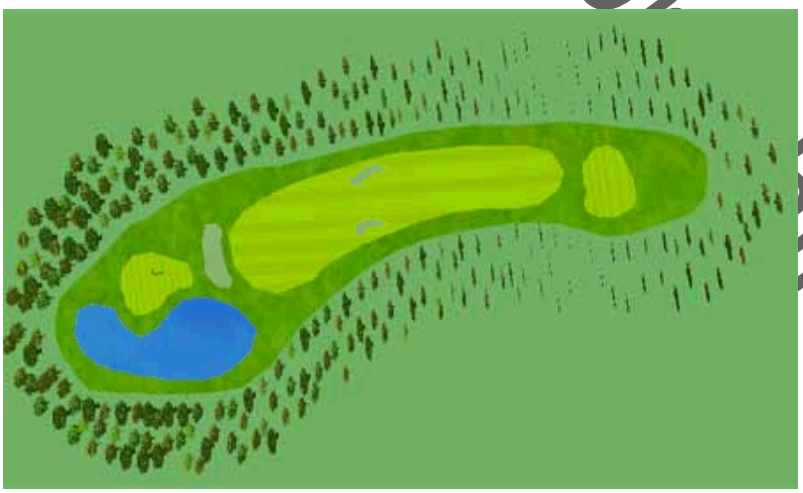
Golf Course 02

Tee low view point

Figure 50: Golf course 02 – Tee low view

Tee high view point

Figure 51: Golf course 02 – Tee high view



Bird eye view

Note: the golf course view is rotated 90 degrees from the original viewpoint above

Figure 52: Golf course 02 – Bird eye view

Golf Course 03

Tee low view point

Figure 53: Golf course 03 – Tee low view

Tee high view point

Figure 54: Golf course 03 – Tee high view

Bird eye view

Note: the golf course view is rotated 90 degrees from the original viewpoint above

Figure 55: Golf course 03 – Bird eye view

Unofficial Document

Appendix E

Golf course Multiple point views

There are seven predefined view points on both Golf courses 01 and 03. Both golf courses contain the same *predefined view points.* On that note, a set of seven screen shots will be produced.

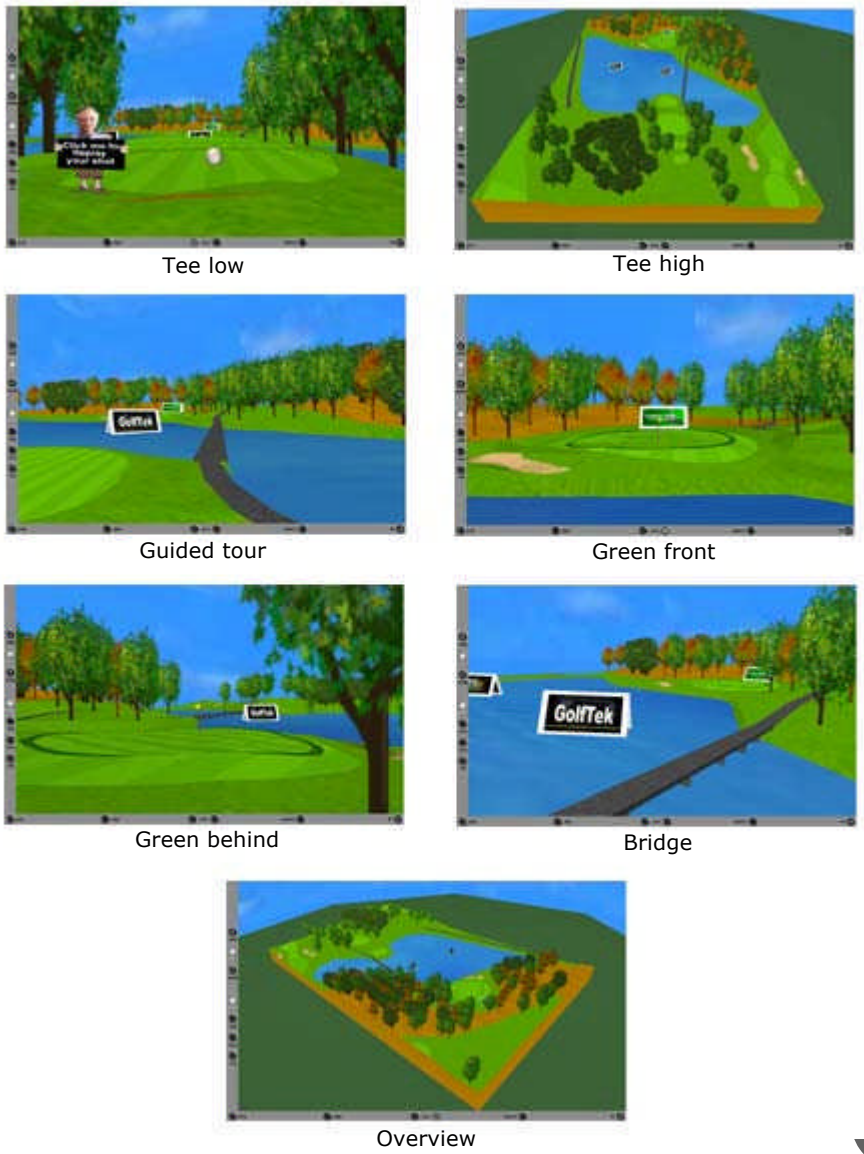


Figure 56: Multiple view points

Figure 57: Mr. Norman Lindsay sign pole

Appendix F

Test Results (screen shots)

Below are some of the testing screen shots taken

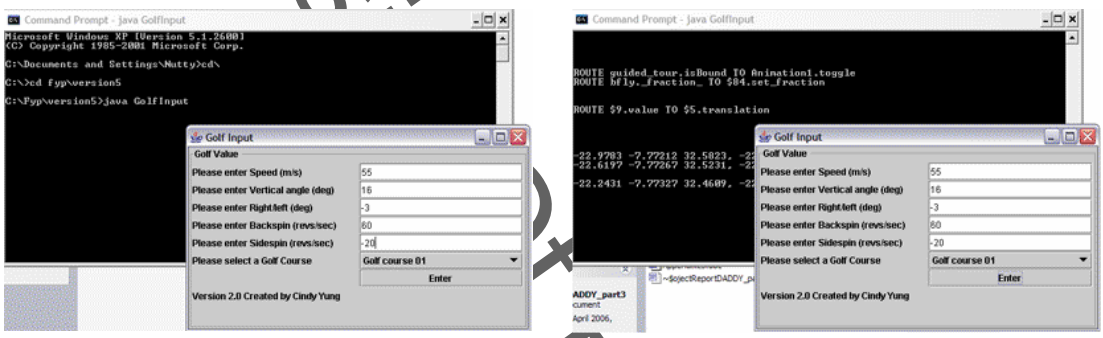


Figure 58: Insert values in all text fields on 'Golf course 01'. Test on submission of values.

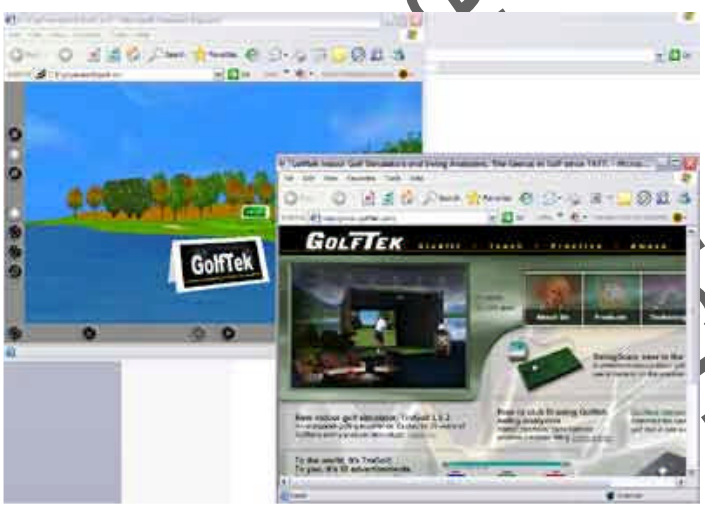


Figure 59: Test if 'GolfTek' billboard open

'GolfTek' web sites in a new window

University of Hertfordshire

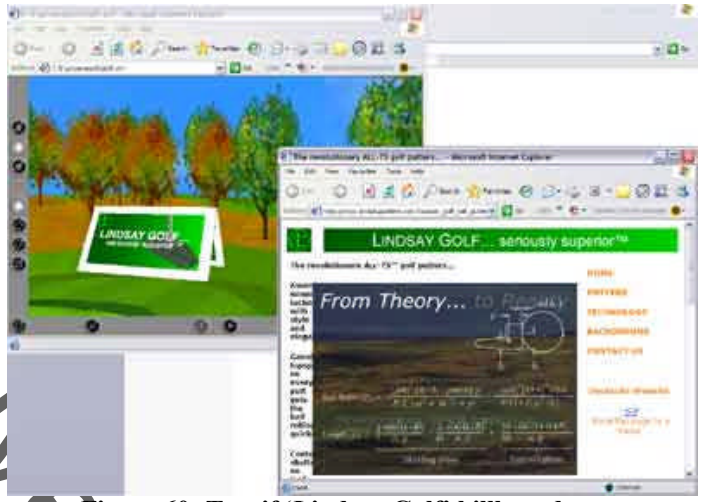


Figure 60: Test if 'Lindsay Golf' billboard open 'Lindsay Golf' web sites in a new window

Appendix G

User Manual

This User manual assumes readers are reading from the 'ReadMe.htm' which is located in the CD enclosed.

How to install and run '3D Golf Ball Trajectory' application

The first thing you need to do is to download the application file, which is located below.

Please select a version :

View from University (u:\) or **View from Home/Work** (c:\)

A new window will open up (Figure 1) asking "Do you want to save or open this file?" Please select **'Save'** button.

University Of Hertfordshire

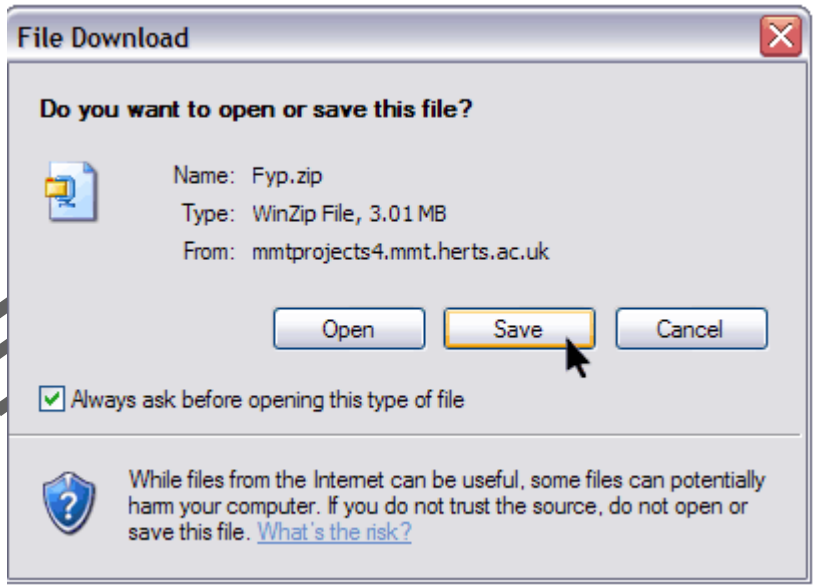

File Download
Do you want to open or save this file?
Name: Fyp.zip
Type: WinZip File, 3.01 MB
From: mmtprojects4.mmt.herts.ac.uk
Open
Save
Cancel
Always ask before opening this type of file
While files from the Internet can be useful, some files can potentially harm your computer. If you do not trust the source, do not open or save this file. What's the risk?

Figure 1: Please select **Save** button

Save 'Fyp.zip' file onto **Desktop** (Figure 2)

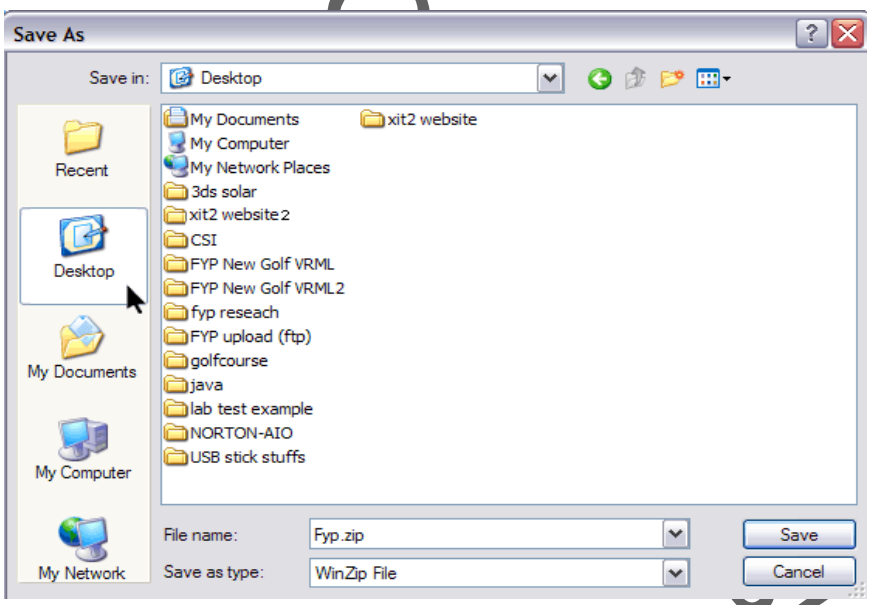


Figure 2: Saving file on **Desktop**

Once download is completed, please select **'Close'** button (Figure 3)

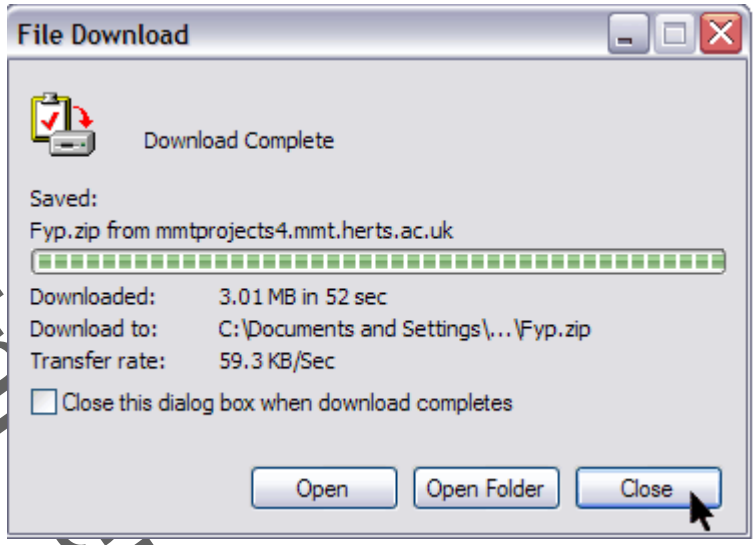


Figure 3: Download Completed

Now, locate the '**Fyp.zip**' file on your Desktop and right click, select **'WinZip\Extract to here'** (Figure 4)

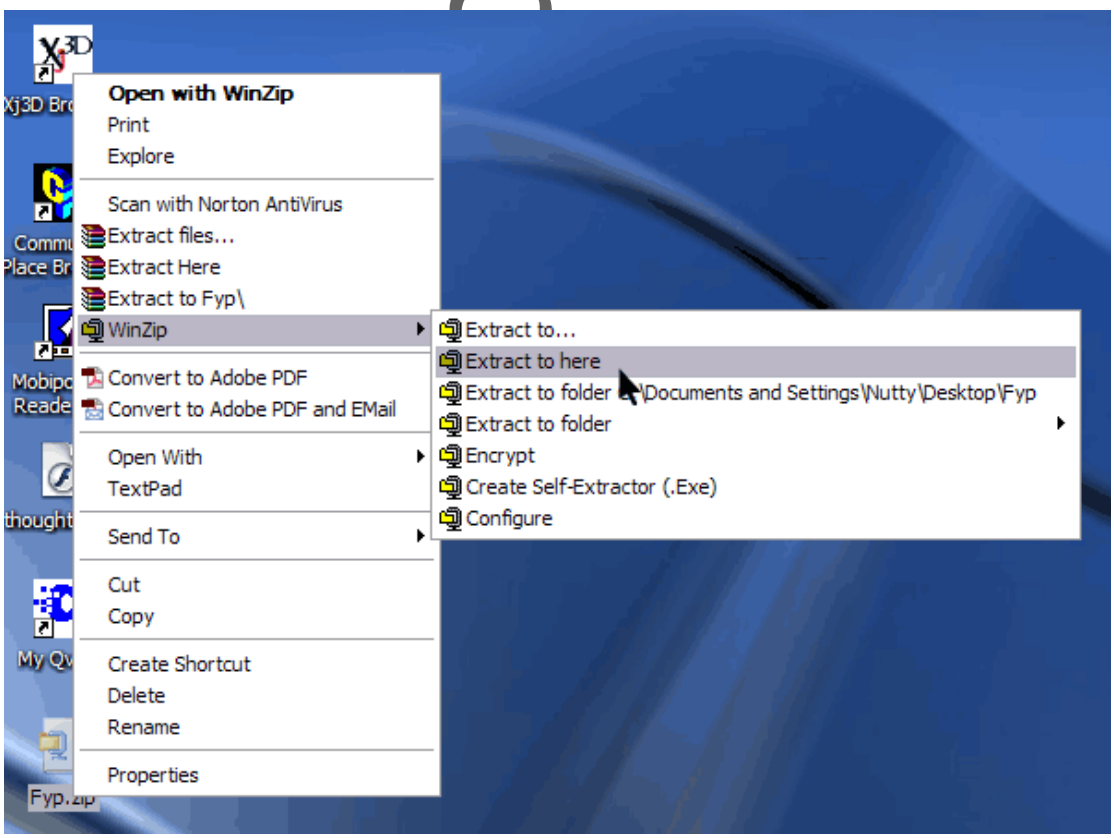


Figure 4: Extracting **'Fyp.zip'** file on to desktop

A folder called **'Fyp'** will be placed on the Desktop.

Next is to install **VRML browser**

Double click on **Fyp** folder inside you will see two items, **'version5'** folder and **'cortvrml.exe'** (Figure 5)

Double click on **cortvrml.exe** to start the installation and follow on screen instruction. (if cortvrml.exe' does not work on your machine, you can download from a third party website.)

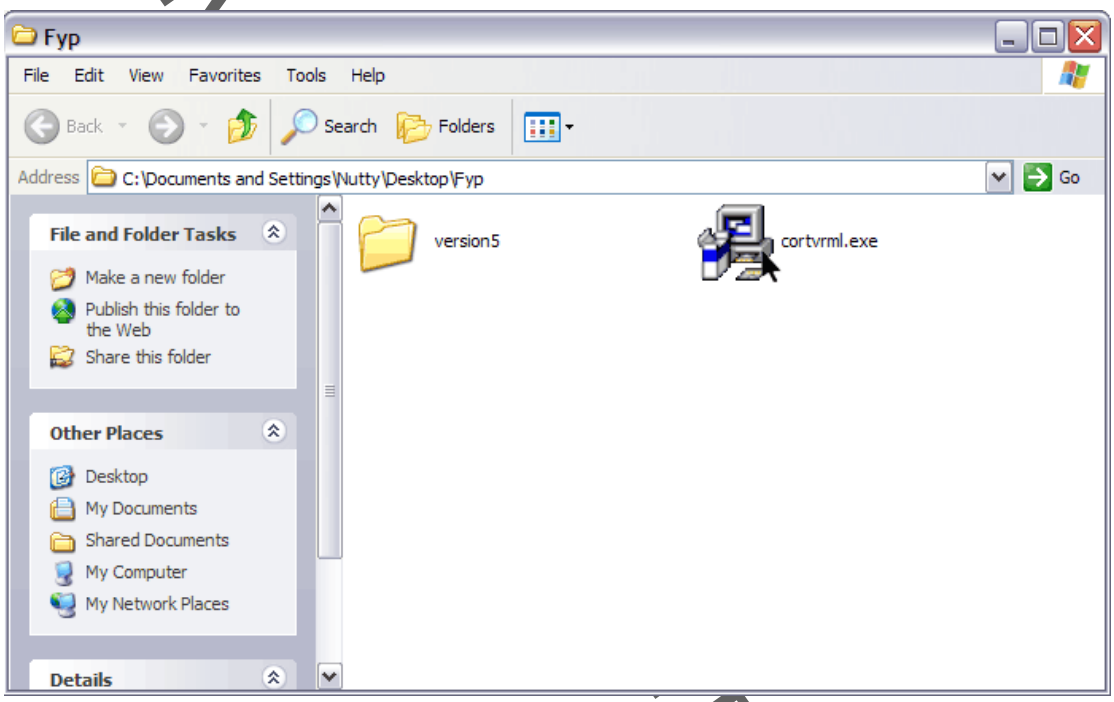


Figure 5: Installing **VRML browser**

Once the VRML Browser is installed, right click on **Fyp** folder and select **Cut** (Figure 6) and **Paste** folder into your current **Drive**. (I will put my into C:\) (Figure 7)

University of Hertfordshire

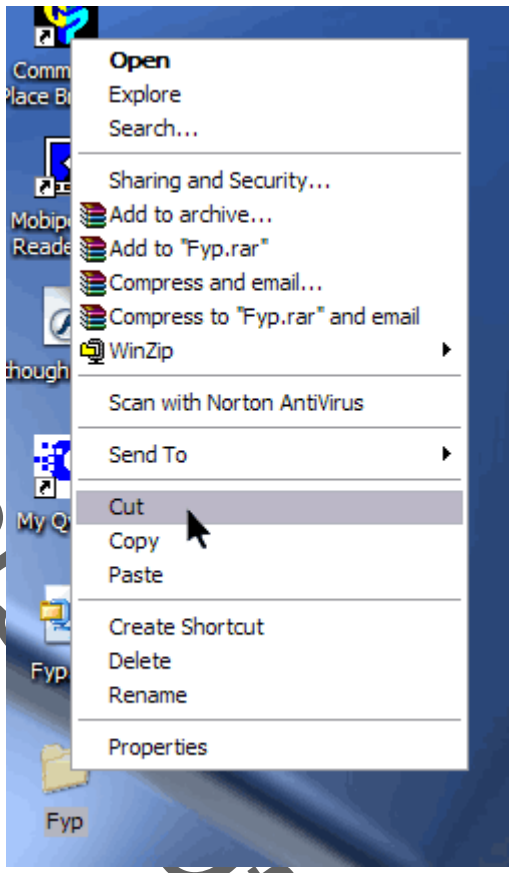


Figure 6: Cutting **Fyp** folder

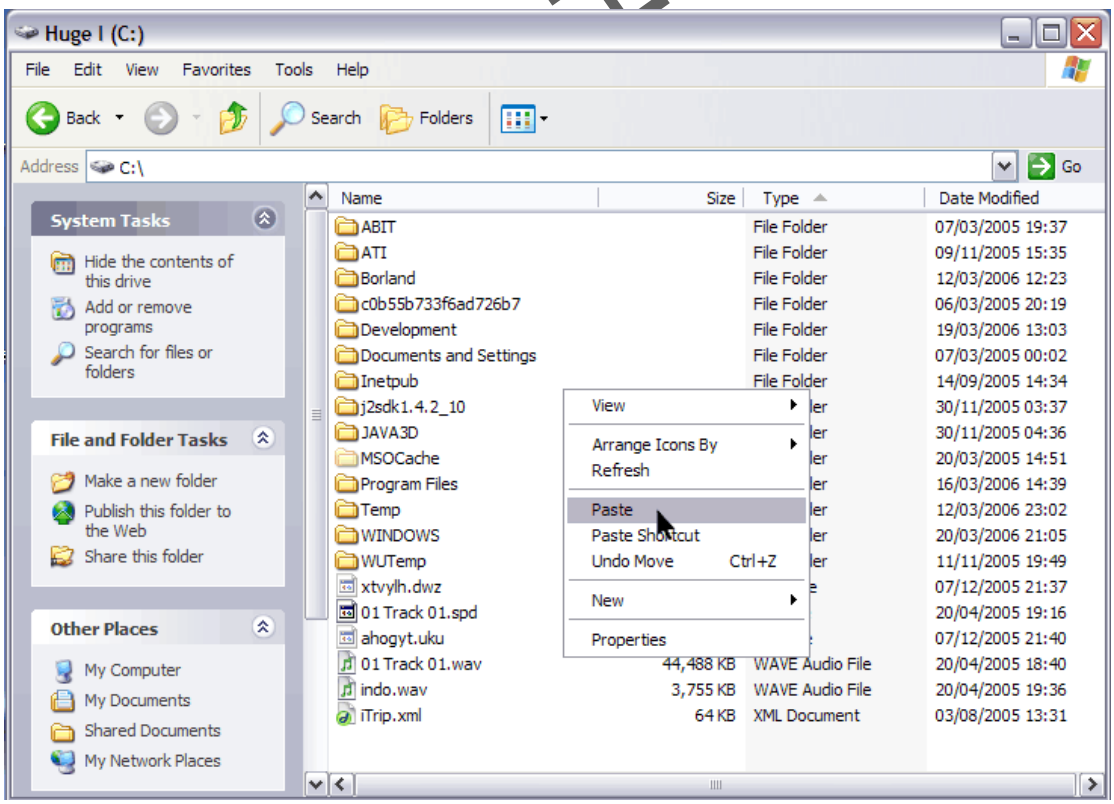


Figure 7: Paste **Fyp** folder into **C:**

Now the Fyp folder is pasted in **C:** you are ready to run the application.

Open **'Command Prompt'** which is located on 'Start \Run'.

Type **'cmd'** in the text field (Figure 8) and seclect **'OK'** button

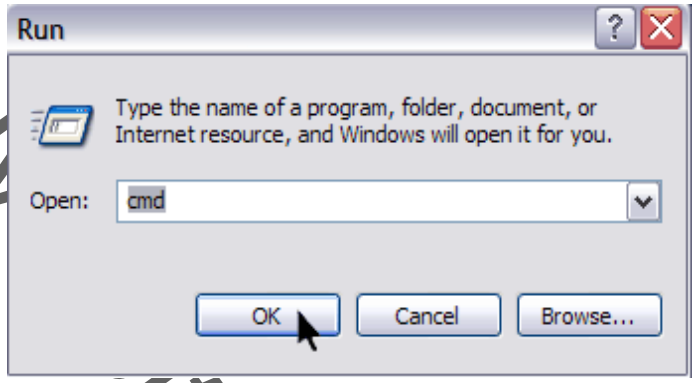


Figure 8: Type **cmd** into text field

University Of Hertf

Command Prompt should had appeared.

Type the following in Command Prompt (Do not skip ahead):

- **cd** 'Press Enter ' (Figure 9)
- **cd fyp\version5** 'Press Enter' (Figure 10)
- **javac GolfInput.java** 'Press Enter' (Figure 11)
- **java GolfInput** 'Press Enter' (Figure 12)

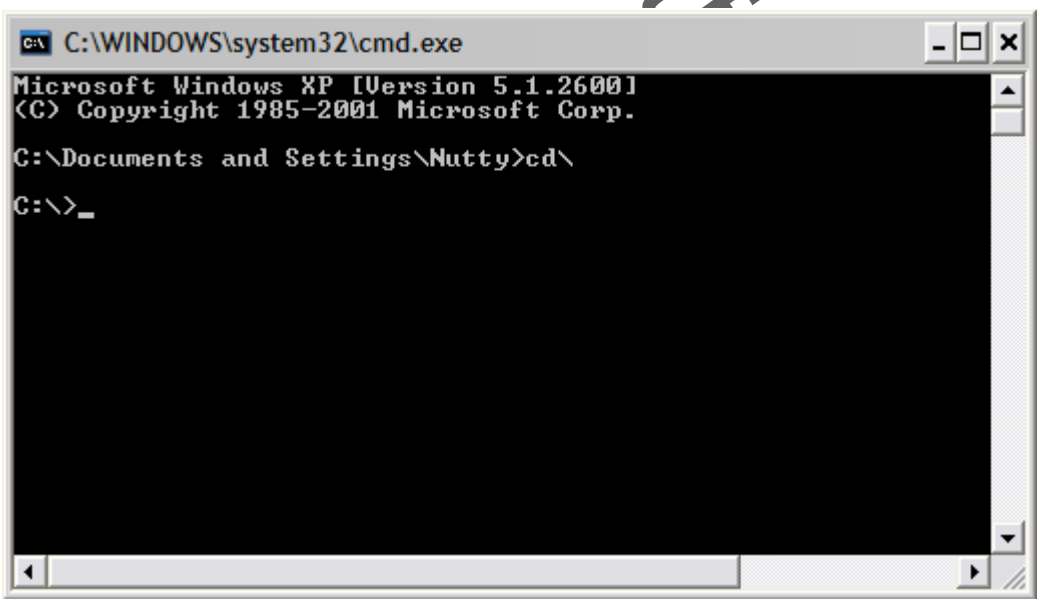


Figure 9: Type **cd** in Command Prompt

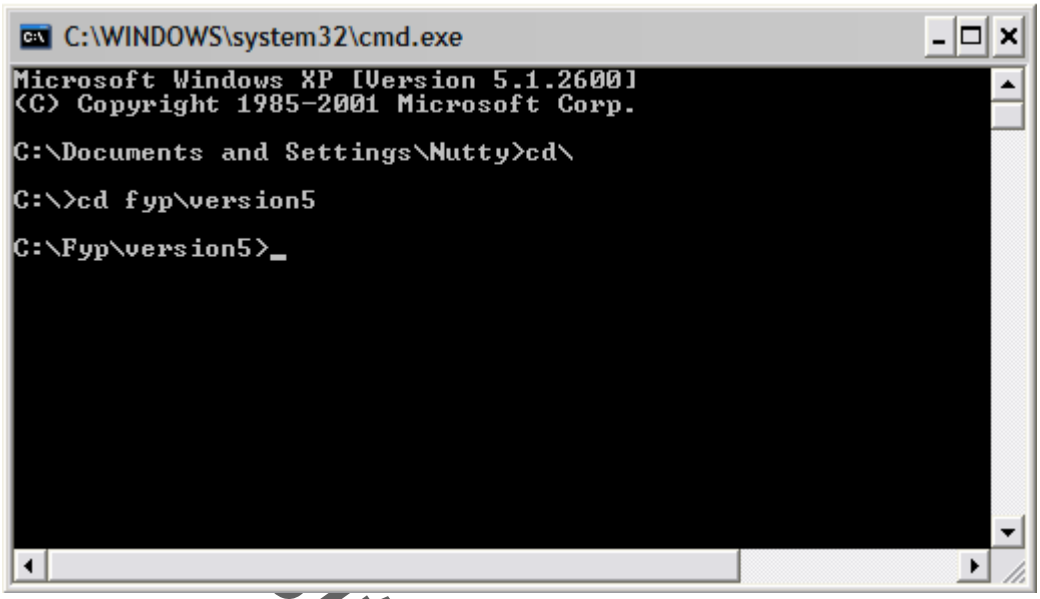


Figure 10: Type **cd fyp\version5** in Command Prompt

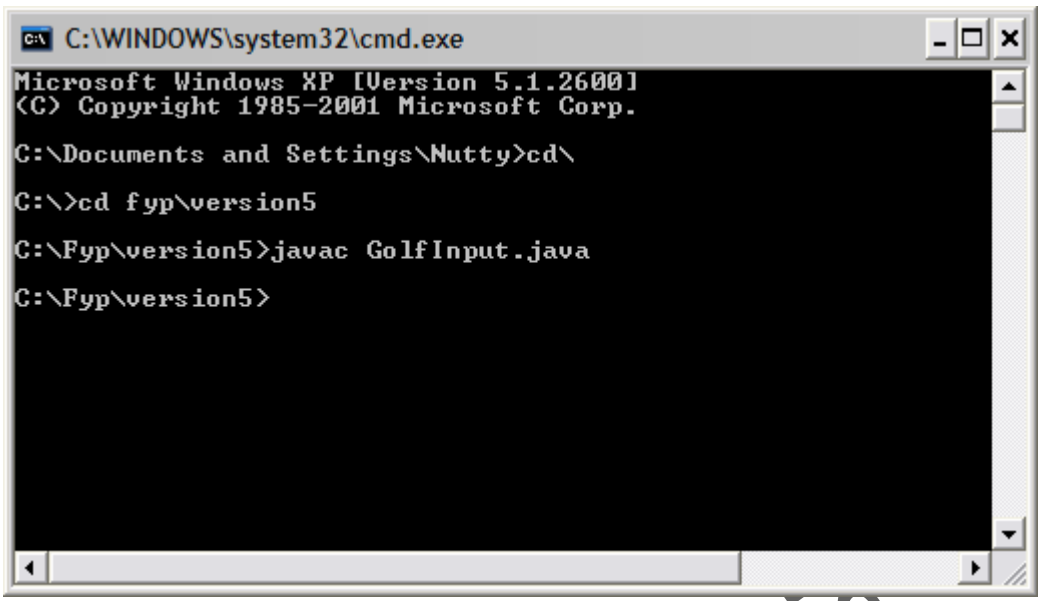


Figure 11: Type **javac GolfInput.java** in Command Prompt

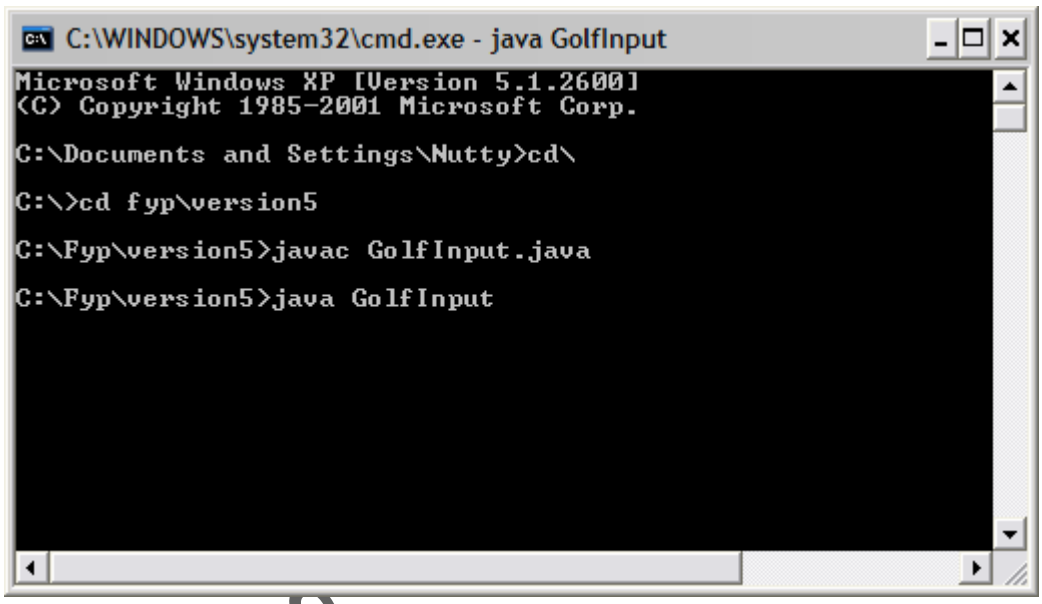


Figure 12: Type **java GolfInput** in Command Prompt

A java window (Golf Input) should had appear on your screen (Figure 13). Type in any values and select **'Enter'**

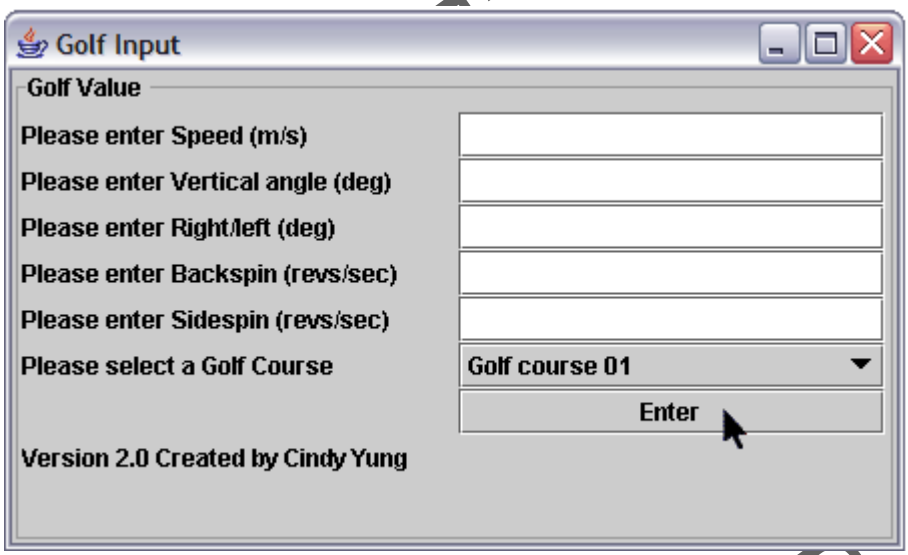


Figure 13: **3D Golf Ball Trajectory** application

Once Command Prompt stop running, go back to **C:\Fyp\version5** inside you should see a file called 'Golf.wrl' double click on the file. This will open in Internet Explorer.

To change Golf Courses, simply click the drop down list 'select a golf course' on the

sity Of hire

java application (GolfInput) and click enter. Lastly click on the 'Refresh' button on the Internet Explorer. This will display the newest result

You have now **Successful** installed and running the **'3D Golf Ball Trajectory'** application.

How to navigate around VRML browser

Information from third party

Using Viewpoints A viewpoint describes a position and orientation for viewing the scene. The VRML author will probably want to guide the user to the best vantage points for viewing it. Not all worlds contain viewpoints, but when they do, you can use them.

To activate a viewpoint, do one of the following:

-

Click the VIEW in the horizontal toolbar or choose viewpoints from the pop-up menu, and then select a viewpoint from the list of the predefined viewpoints.
- Click the arrow buttons to move to the next or previous viewpoint. You can also press Page Down or Page Down keyboard button.

Note:

If there are no predefined viewpoints in a world, the Empty message appears.

Moving around: Walk, Fly, and Study

There are three main navigation modes that Cortona VRML Client offers: WALK, FLY, and EXAMINE. You can switch the navigation mode by clicking buttons on the vertical toolbar (click STUDY to enter EXAMINE mode). Each navigation mode may have several options: PLAN, PAN, TURN, and ROLL. The combination of navigation mode and its option determines the possible camera motion and its orientation. Please note that the VRML author has an opportunity to specify which navigation paradigm should be used in the scene by default. Some worlds don't allow

University Of Hertfordshire

the user to use navigation controls, but they may provide on-screen cues to navigation.

You can navigate with the mouse, the keyboard, or both mouse and keyboard. To move around a 3D world using the mouse:

1. Choose a navigation mode.
2. Position the pointer anywhere in the 3D window and press the left mouse button.
3. Move the mouse while holding down its left button. The *direction* in which you drag the mouse determines the camera motion.
4. Release the left mouse button to stop moving.

Note: The *distance* that you drag the mouse determines the *speed* with which the camera moves. If you stop moving the mouse, the camera will continue moving until you release the mouse button. To accelerate the camera's movement or rotation, press SHIFT, CTRL or SHIFT+CTRL.

and

Use WALK+PLAN to move in a horizontal plane.

Forward - move closer

Backward - move further

Right - turn to the right

Left - turn to the left

Note: Move the mouse forward or backward while holding down the Space key to turn upward, downward.
Move the mouse left or right while holding down the ALT key to move left or right.

University Of Hertfordshire

and

Use WALK+PAN to move left or right in a horizontal plane.

Forward - move closer

Backward - move further

Right - move right

Left - move left

Note: When you move the camera right or left, the world will appear to move in the opposite direction.

and

Use WALK+TURN to change the angle of the camera in a world.

Forward - turn upward

Backward - turn downward

Right - turn to the right

Left - turn to the left

and

Use FLY+PLAN to move left or right.

Forward - move the camera forward towards its longitudinal axis

Backward - move the camera backward

Right - turn the camera to the right around its vertical axis

University Of Hertfordshire

Left - turn the camera to the left around its vertical axis

Note: The camera's vertical axis may be inclined in a 3D Space.
Move the mouse while holding down the ALT key to switch FLY+PAN.
Move the mouse while holding down the Space key to switch FLY+TURN.
Move the mouse while holding down the ALT+Space keys to switch FLY+ROLL.

and

Use FLY+PAN to move up, down, left, or right within a single vertical plane.

Forward - move up

Backward - move down

Right - move right

Left - move left

Note: When you move the camera, the world will appear to move in the opposite direction.

and

Use FLY+TURN to turn the camera.

Forward - turn the camera upward around its horizontal axis

Backward - turn the camera downward around its horizontal axis

Right - turn the camera to the right around its vertical axis

Left - turn the camera to the left around its vertical axis

University Of Hertfordshire

and

Use FLY+ROLL to incline the camera.

Right - incline to the left

Left - incline to the right

and

Use STUDY+PLAN to examine an object from various angles.

Forward - move the camera forward

Backward - move the camera backward

Right, Left - move the camera around the central point which is defined by the center of bounding box of the geometry in the 3D scene.

and

Use STUDY+TURN to examine an object from various angles.

Forward, Backward, Right, and Left - move the camera around the central point which is defined by the center of bounding box of the geometry in the 3D scene.

Note: Move the mouse while holding down the ALT key to switch STUDY+PAN.
Move the mouse while holding down the Space key to switch STUDY+PLAN.
Move the mouse while holding down the ALT+Space keys to switch STUDY+ROLL.

University Of Hertfordshire

and

Use STUDY+ROLL to incline the camera around the central point which is defined by the center of bounding box of the geometry in the 3D scene.

Right - incline to the left

Left - incline to the right

Note: The VRML author can choose the rotation center in the EXAMINE navigation mode. By default, when in EXAMINE mode, Cortona VRML Client sets the center of rotation in the center of the bounding box of the scene geometry. But it is possible to change this center by specifying three additional values in the avatarSize field of the NavigationInfo node. According to the VRML specification, the first three values are the avatar dimensions, while additional values may be used for browser-specific purposes. In Cortona VRML Client the fourth, fifth and sixth values specify the center of scene rotation in the world coordinates X,Y and Z.

Use GOTO to move close to object in a world. Select GOTO in the toolbar and then click on an object in the world. You'll move directly to it.

Restore, Fit, and Align

Cortona VRML Client provides three mechanisms that can help to re-orient a camera if you have lost your way in a world. Unlike the navigation tools, these buttons invoke predefined actions that take place as you click on them.

Use RESTORE to automatically return to the loaded world's original active viewpoint.

University Of Hertfordshire

Use FIT to make the scene view fully visible in the Cortona VRML Client 3D window.

Use ALIGN to position the camera's horizontal and longitudinal axes parallel to the scene horizontal plane.

Using Keyboard for Navigating

You can control the camera using keyboard commands. The functionality for buttons corresponds to the movement of your mouse and depends on the navigation type, its option, and the specified skin. Please note that the following description of keyboard commands is presented for the FLY+PLAN navigation and the Default skin.

- Arrow Up - move closer.
- Arrow Down - move further away.
- Arrow Right - turn to the right.
- Arrow Left - turn to the left.
- Arrow Up on the numeric keypad - move closer.
- Arrow Down on the numeric keypad - move further.
- Arrow Right on the numeric keypad - move to the right. If the STUDY mode - move the camera around the center of rotation.
- Arrow Left on the numeric keypad - move to the left. If the STUDY mode - move the camera around the center of rotation.
- 7 on the numeric keypad - turn the camera downward around its horizontal axis.
- 9 on the numeric keypad - turn the camera upward around its horizontal axis.
- 1 on the numeric keypad - incline to the right. Note: It is not available in WALK mode.
- 3 on the numeric keypad - incline to the left. Note: It is not available in WALK mode.
- Gray Plus - move up in the case of FLY mode.

University Of Hertfordshire

- Gray Minus - move down in the case of FLY mode.
- 1 on the alphanumeric keyboard - incline to the right. Note: It is not available in WALK mode.
- 2 on the alphanumeric keyboard - incline to the left. Note: It is not available in WALK mode.
- 3 on the alphanumeric keyboard - turn downward. If the STUDY mode - move the camera around the center of rotation.
- 4 on the alphanumeric keyboard - turn upward. If the STUDY mode - move the camera around the center of rotation.
- Page Down - next viewpoint.
- Page Up - previous viewpoint.

Pause - Enables or disables CylinderSensor, PlaneSensor, SphereSensor, and TouchSensor in Cortona VRML Client.
To accelerate the camera's movement or rotation: Press SHIFT, CTRL or SHIFT+CTRL and one of the above keys simultaneously.
The ALT and SPACE keys allow the user to quickly change the navigation option to PAN and TURN accordingly in the FLY navigation mode (to ROLL and PLAN in the STUDY mode). The ALT and SPACE keys simultaneously pressed activate ROLL in the FLY navigation mode or PAN in STUDY.

Interacting with the Scene

Some of the objects in the scene may provide special effects that allow the user to interact with the scene in different ways. As you position the pointer over the object, containing a VRML *sensor*, the pointer changes:

- Touch Sensor. Detects a click or the pointer over the object. The sort of activity in the scene is decided by the VRML author.
- Anchor. Clicking will link to the other object, VRML world or HTML document.

University Of Hertfordshire

- Cylinder Sensor. Transforms pointer motion into a rotation of the object around its axis.
- Sphere Sensor. Transforms pointer motion into a rotation of the object around its two axes.
- Plane Sensor. Transforms pointer motion into a moving of the object.
- Drop Sensor (VRML extension). Handling a drag-and-drop operation. Retrieves an object's uniform resource locator (URL) of an object (resource) dragged to the 3D window.

University Of Hertfordshire

Appendix H

Questionnaire

1. What do you think of the overall simulation?

Excellent Good Average Poor Others

2. What do you think of the look and effect of the interface?

Excellent Good Average Poor Others

3. Does the 3D golf course match up in the real world?

Yes No

4. How would you feel if instruction is added in the interface? Would this provide better usage?

Yes No

5. By adding a background image or a company logo, does it provide a professional look and identity?

True False

6. Is it easy to use the VRML browser functions? (ie: navigate around the golf course)

Yes No

7. What do you think of the images used in the VRML Golf course?

Excellent Good Average Poor Others

8. Are you happy with the simulation trajectory of a golf ball?

Yes No

9. Would you like to have more golf courses to choose from?

Yes No

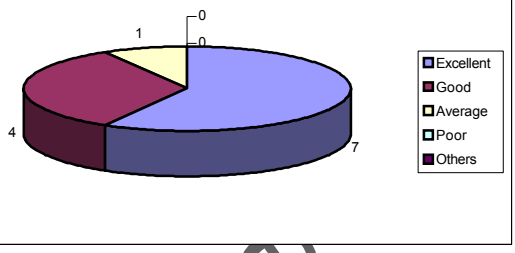
10. Would you like to access the simulator through the Internet?

Yes No

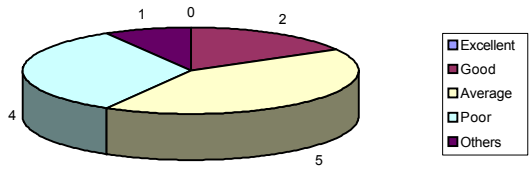
University Of Hertfordshire

Questionnaire results

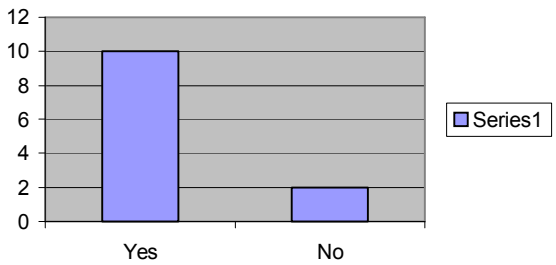
1. What do you think of the overall simulation?



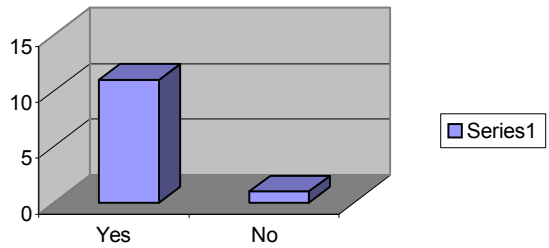
2. What do you think of the look and effect of the interface?



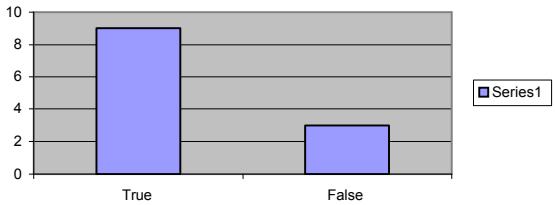
3. Does the 3D golf course match up in the real world?



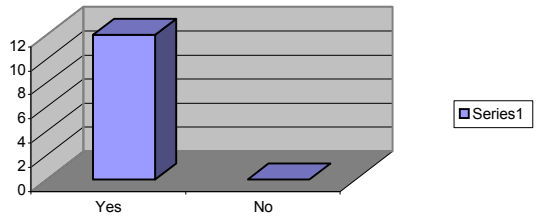
4. How would you feel if instruction is added in the interface? Would this provide better usage?



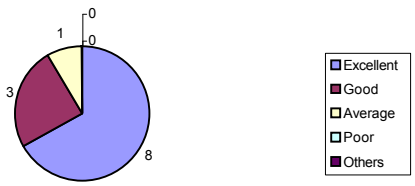
5. By adding a background image or a company logo, does it provide a professional look and identity?



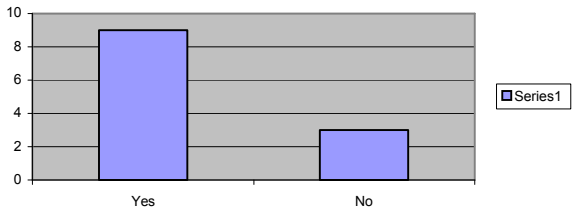
6. Is it easy to use the VRML browser functions? (ie: navigate around the golf course)

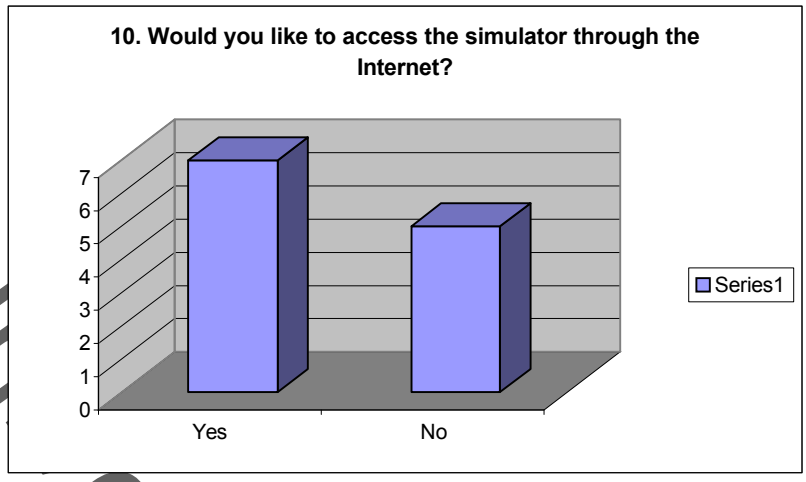


7. What do you think of the images used in the VRML Golf course?



9. Would you like to have more golf courses to choose from?





University Of Hertfordshire

Appendix I

Story board

Interface designs

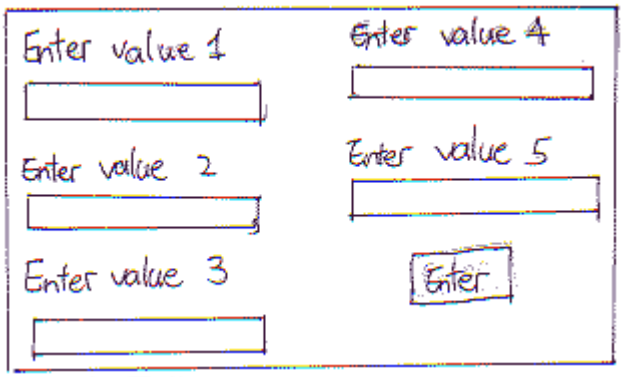


Figure 61: Interface design 01

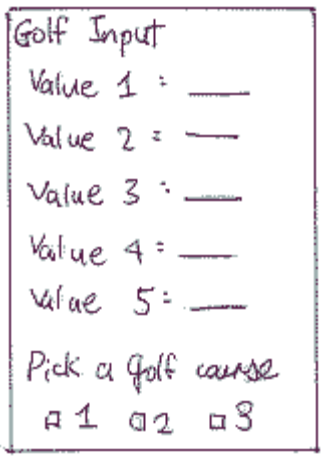


Figure 62: Interface design 02

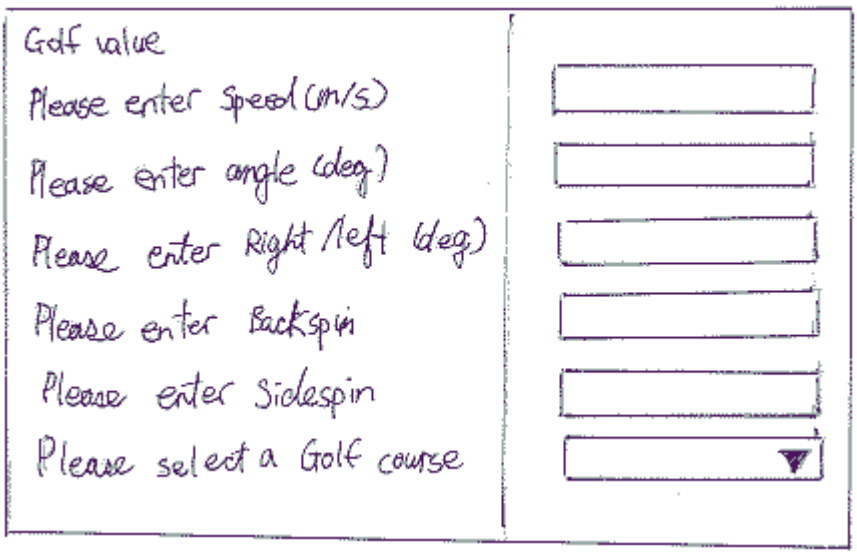


Figure 63: Interface design 03

Golf course designs

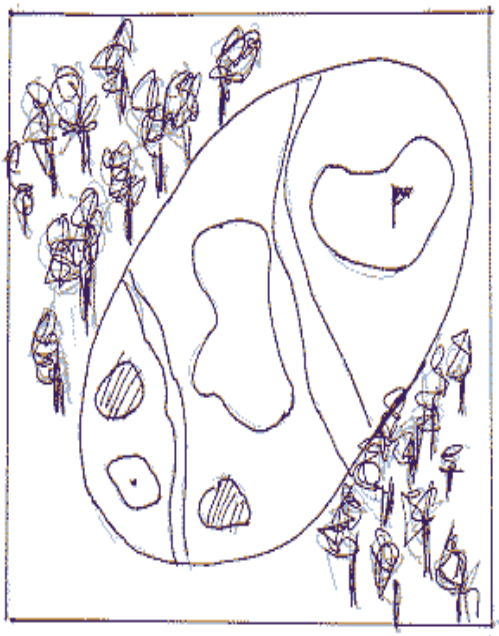


Figure 64: Golf course design 01

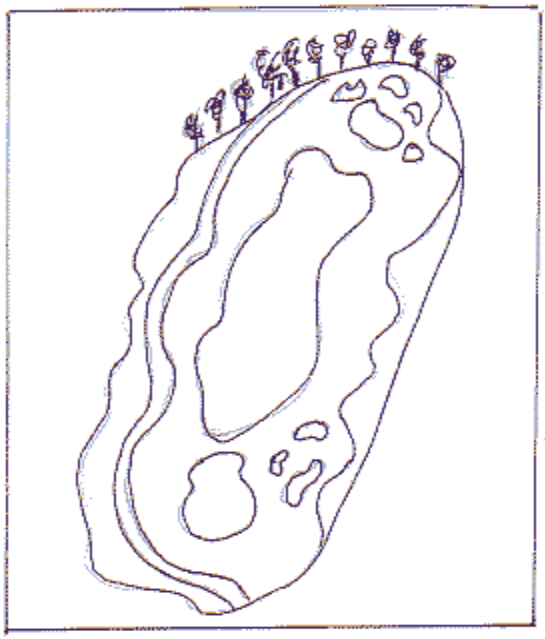


Figure 65: Golf course design 02

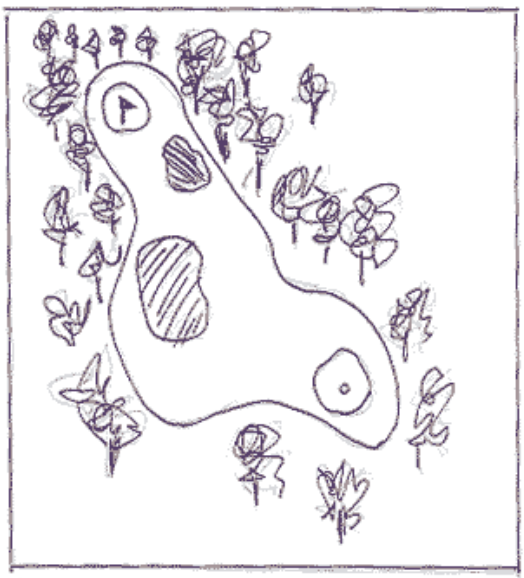


Figure 66: Golf course design 03

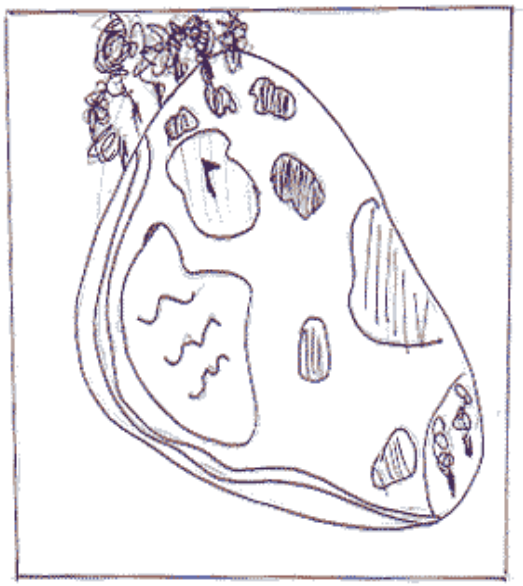


Figure 67: Golf course design 04

Golf course design 03

Appendix J

Project Time Plan – Gantt Chart

Initial time plan was made optimistically as some tasks were not carried out and some amendments were made in the process of the projects.
As a result, an updated version of the time plan is produced which is accompanied with the original one.

University Of Hertfordshire

Initial Project Time Plan Gantt Chart

	Task Name	Duration	Start	Finish
1	Research	2 mons	Fri 14/10/05	Thu 08/12/05
2	To research various 3D application programming language on their compatibility. The main objective is to	1 mon	Fri 14/10/05	Thu 10/11/05
3	Research in-depth on 3D Java language	1 mon	Fri 11/11/05	Thu 08/12/05
4	Design Development	14 wks	Tue 22/11/05	Thu 23/02/06
5	To create a 3D sphere/golf ball with animation to move from left to right.	1 wk	Tue 22/11/05	Mon 28/11/05
6	Testing	2 days	Tue 29/11/05	Wed 30/11/05
7	Stage write up	1 day	Thu 01/12/05	Thu 01/12/05
8	To create a 3D spinning sphere/ golf ball travelling in a straight line in a 3D	2 wks	Fri 02/12/05	Thu 15/12/05
9	Testing	2 days	Fri 16/12/05	Mon 19/12/05
10	Stage write up	1 day	Tue 20/12/05	Tue 20/12/05
11	Creating a 3D environment showing a sphere/golf ball travelling in a trajectory form	2 wks	Wed 21/12/05	Tue 03/01/06
12	Testing	2 days	Wed 04/01/06	Thu 05/01/06
13	Stage write up	1 day	Fri 06/01/06	Fri 06/01/06
14	To create zoom, bird eye view and landing area functions for object 2-4	3 wks	Sat 07/01/06	Thu 26/01/06
15	Testing	2 days	Fri 27/01/06	Mon 30/01/06
16	Stage write up	1 day	Tue 31/01/06	Tue 31/01/06
17	Display of a golf ball travelling in different tur	2 wks	Wed 01/02/06	Tue 14/02/06
18	Testing	2 days	Wed 15/02/06	Thu 16/02/06
19	Stage write up	1 day	Fri 17/02/06	Fri 17/02/06
20	Implement trajectory formula from research into the 3D golf course programme	3 wks	Sat 18/02/06	Thu 09/03/06
21	Testing	2 days	Fri 10/03/06	Mon 13/03/06
22	Stage write up	1 day	Tue 14/03/06	Tue 14/03/06
23	Testing	7 wks	Wed 15/03/06	Tue 02/05/06
24	Final testing	3 wks	Wed 15/03/06	Tue 04/04/06
25	Final write up and presentation	1 mon	Wed 05/04/06	Tue 02/05/06

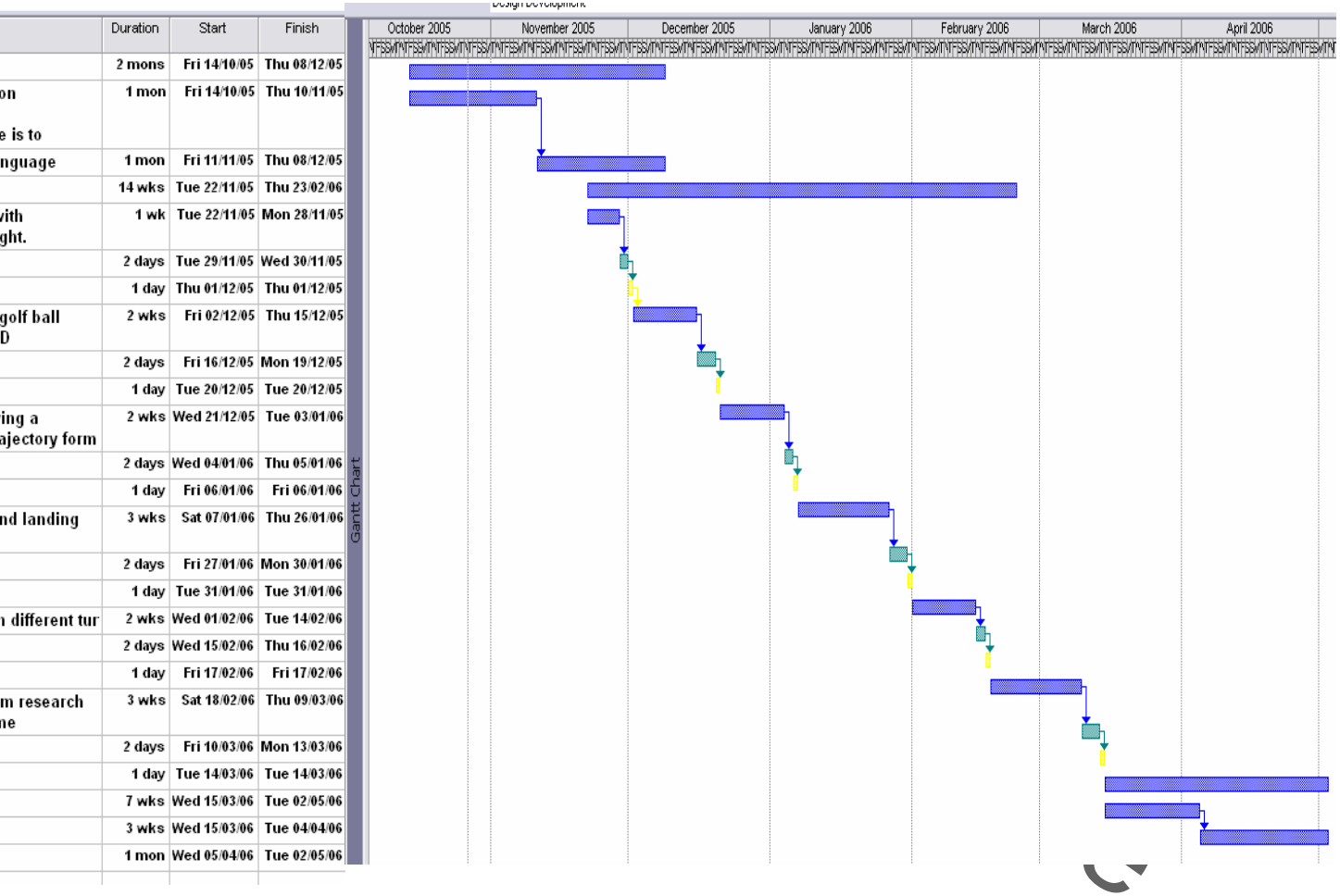
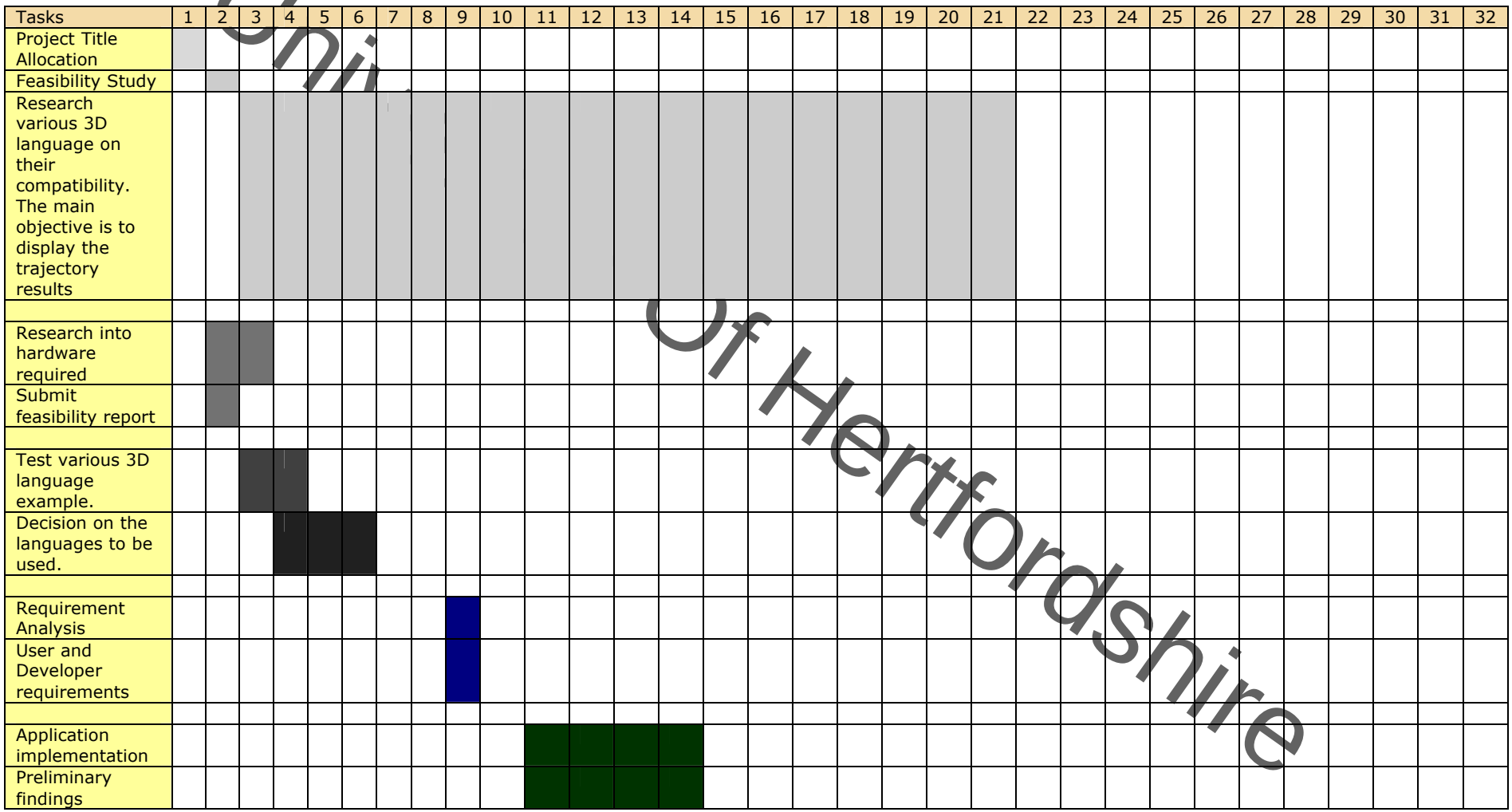
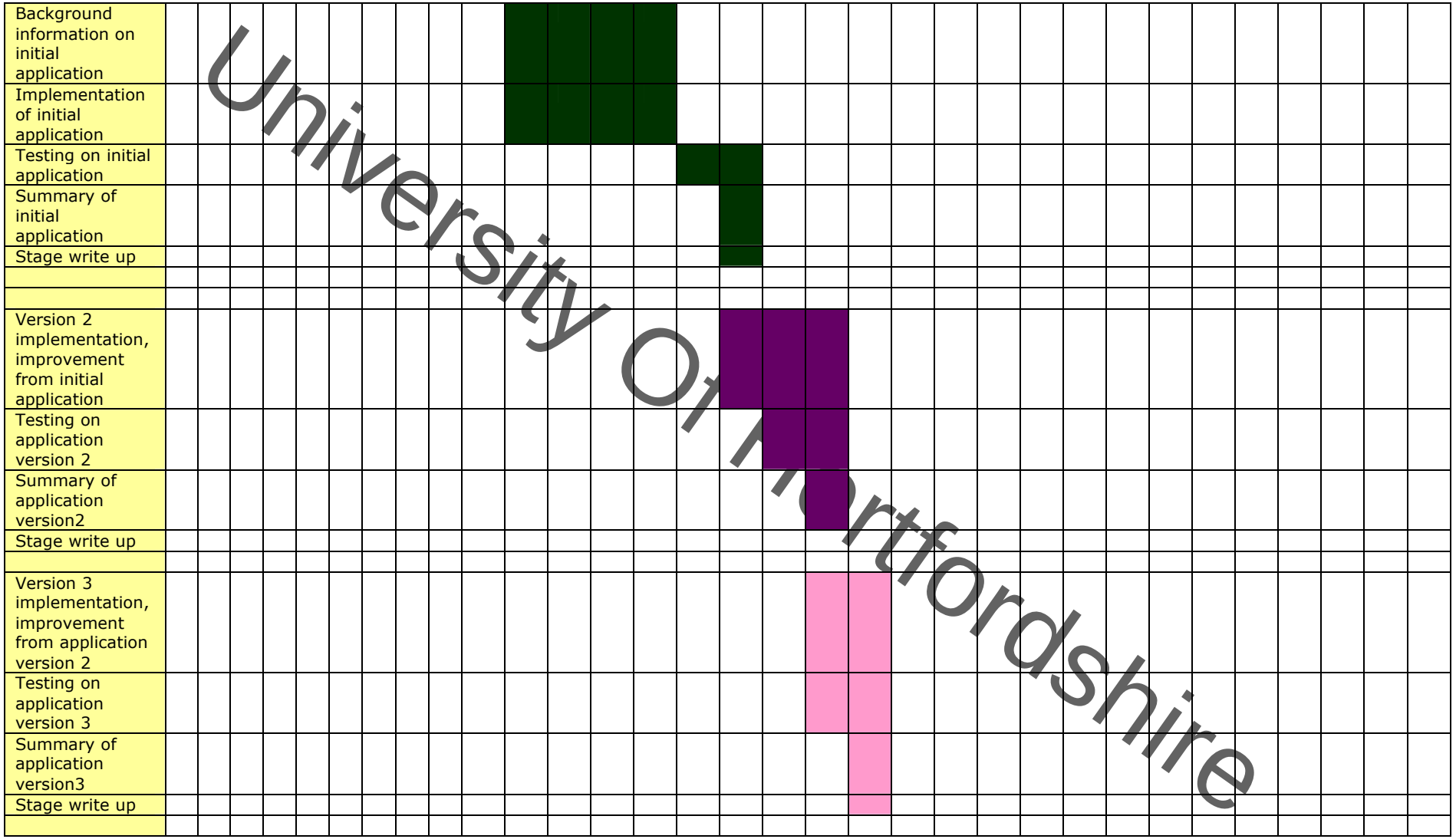


Figure 68: Initial Gantt Chart

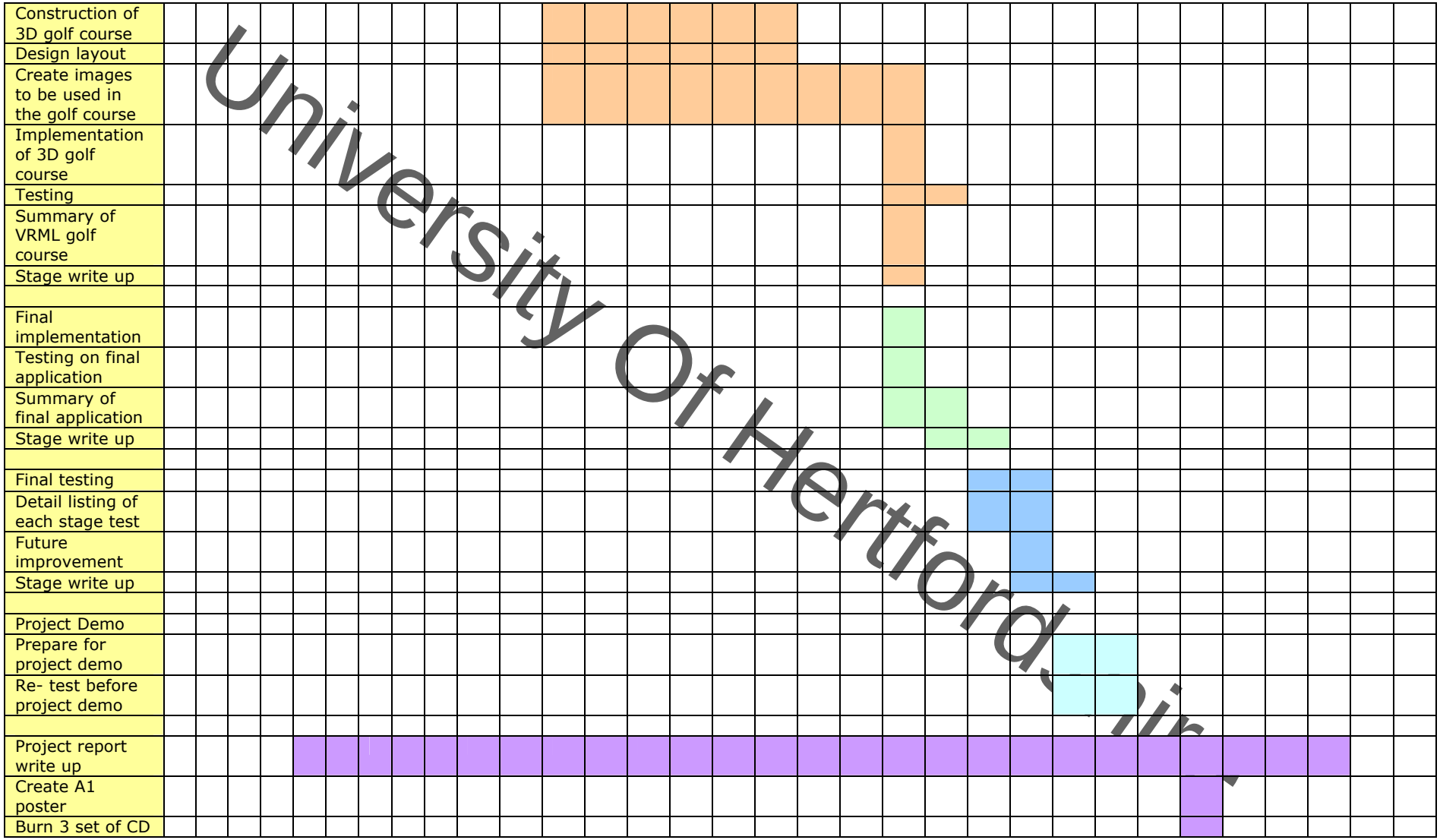
Final Project Time Plan Gantt Chart



University Of Hertfordshire


Background information on initial application
Implementation of initial application
Testing on initial application
Summary of initial application
Stage write up
Version 2 implementation, improvement from initial application
Testing on application version 2
Summary of application version2
Stage write up
Version 3 implementation, improvement from application version 2
Testing on application version 3
Summary of application version3
Stage write up

University Of Hertfordshire



University Of Hertford

containing, report, application and other related information																																
Prepare for poster session																																
Poster sessions																																

University Of Hertfordshire

Appendix K

CD-ROM Contents

Application

- classes
- src
- images

Final Report

Poster

- posterUni.ppt
- Poster.JPG

Software

- Cortvrml.exe (VRML Browser)

ReadMe.htm

- Fyp.zip
- FypUni.zip
- img

University Of Hertfordshire